TRABAJO DE FINAL DE CARRERA

TÍTULO:	**Engine and Fuel Manager System for Unmanned Aerial Vehicles**
AUTOR:	**Julio Sagardoy Pérez**
TITULACION:	**Ingeniería Técnica de Telecomunicaciones, especialidad en Sistemas de Telecomunicación**
DIRECTOR:	**Enric Pastor Llorens**
FECHA:	**21 de Mayo de 2009**

Título: Engine and Fuel Manager System for Unmanned Aerial Vehicles

Autor: Julio Sagardoy Pérez

Director: Enric Pastor Llorens

Fecha: 21 de mayo de 2009

Resumen

Controlar el estado de los distintos sistemas que forman parte de un motor de combustión interna deja de ser una opción cuando éste está dando la fuerza motriz a un UAV. Ello se debe a la fatalidad que supondría un posible paro del motor en pleno vuelo por culpa de un mal funcionamiento del mismo debido a factores externos, como por ejemplo, cambios en la densidad del aire producidos por cambios de altura de vuelo que modifiquen la combustión; o internos: depósitos de carburante vacíos, fugas… Esto es perfectamente evitable si se dispone de un sistema que permita la monitorización y diagnóstico del mismo.

A causa de la inexistencia de un servicio que proporcione tal información dentro del proyecto ICARUS, el objetivo de este trabajo será diseñar y desarrollar tal sistema. Un sistema que, además de controlar distintos parámetros del motor específico del UAV Shadow, los transmita luego a la estación base a través de los sistemas de comunicaciones ya existentes o en desarrollo, para finalmente mostrarlos en la terminal de control.

Un sistema de alertas tempranas también estará disponible. Este sistema avisará al supervisor de vuelo en caso de que algún parámetro se encuentre fuera de los rangos nominales. El sistema incluirá también un decisor de prioridades, para evitar un exceso de avisos de alarma, que podrían desconcertar.

La idea inicial es proyectar este servicio como si fuera válido para motores de distintas características que el que equipa el UAV Shadow, pero a la hora de aplicarlo se hará siguiendo las restricciones de dimensiones, peso y rangos de medida que requiera éste.

Un servicio como éste cobra muchísima importancia para asegurar un funcionamiento seguro y fiable de un UAV, y no sólo sirve como diagnóstico continuo del motor en vuelo, sino también como referencia para tener en cuenta antes de tomar cualquier decisión en vuelo.

Title: Engine and Fuel Manager System for Unmanned Aerial Vehicles

Author: Julio Sagardoy Pérez

Director: Enric Pastor Llorens

Date: 21 de mayo de 2009

Overview

Knowing, monitoring and understanding the status of the different systems that form a part of an internal combustion engine becomes a must when it is flying an UAV. An engine cutoff during flight as a consequence of a malfunction due to external issues (for example, air density change because of the height, which changes engine behavior) or internal issues (no fuel, fuel leaking, overheating…) would be avoidable if a monitoring and diagnostics service was being used.

Because of the inexistence of an engine diagnostics service into the UAV Shadow, the main purposes of this project are design and develop it. Furthermore, it will not only supervise its parameters, but transmit them to a ground control station, through the existent (or under development) systems, to be finally shown in the control terminal.

Also an early alarm system will be available in the service. This will warn the supervisor if any of the monitored parameter is out of normal range, or some kind of engine-related failure is expected. An alarm priority system will be also implemented, which intention is to avoid useless alarms which could disturb.

The initial purpose is to develop a compatible with most engines, although it is pretended to be tailored to meet the 250cc UAV Shadow's engine.

Considering a service like is obvious for a reliable operation of an UAV. Not only because the availability of engine diagnostics, but also as a reference to keep in mind before taking any decision during the flight.

CONTENTS

SECTION 1. INTRODUCTION

Knowing, monitoring and understanding the status of the different systems that form a part of an internal combustion engine becomes a must when flying an UAV. An engine cutoff during flight as a consequence of a malfunction due to external issues (for example, air density change because of the height, which changes engine behavior) or internal issues (no fuel, fuel leaks, overheating…) would be avoidable if a monitoring and diagnostics service was available.

Because of the inexistence of an engine diagnostics service into the UAV Shadow, the main purpose of this project is to design and develop it. Furthermore, it will not only supervise its parameters, but transmit them to a ground control station, through the existent (or under development) systems, to be finally shown in the control terminal.

Also an early alarm system will be available in the service. This will warn the supervisor if any of the monitored parameter is out of normal range, or some kind of engine-related failure is expected. An alarm priority system will be also implemented, which intention is to avoid useless alarms which could disturb.

The initial purpose is to develop EFMS compatible with most engines, although it is pretended to be tailored to be installed in the UAV Shadow's.

Considering a service like is obvious for a reliable operation of an UAV, not only because the psychic-related aspects of knowing the engine status, but also as a reference to keep in mind before taking any decision during the flight.

SECTION 2. PROJECT BACKGROUND

2.1 Project inspiration

Being into the Icarus UAS Platform project and helping the ICARUS RESEARCH TEAM came from some time ago. First I have to say that my main purpose was to do Aeronautics Engineering degree, but because of the insane grade needed to get in, it was not possible for me. So I decided to jump into Telecommunications Engineering -electronics and computer hardware also fascinates me-, take some technical baggage and then gradually move by the Aeronautics branch.

But there was something I had clear in mind since I started: the End Degree Project might be in relationship with both aeronautics and telecommunications. By the last year I met Joshua Martinez, who was into the ICARUS RESEARCH TEAM and told me about the things they were performing. I immediately felt attraction for joining them, and told him that I wanted to help out. He referred me to Enric Pastor, who is teacher in computing sciences and is also the chief manager of the ICARUS group. He told me which issues were still pendant, and this one, the Engine and Fuel Manager System (EFMS) caught my attention. I have always been interested in engine mechanics and sensing-purposed hardware and software, so I found this task especially interesting for me.

Because of that, the investigation, development and testing stages of a new engine and fuel monitoring system have been done.

2.2 Context and opportunity

EPSC hosts both Telecommunications and Aeronautics Engineering studies, and has always had a lot of interest in UAV-related issues. That made the EPSC ideal for developing such a complex project as the Icarus UAS Platform is, because it would have a bunch of teachers and students interested in.

Meanwhile, nationally speaking but especially in Barcelona, UAV investigations are eventually taking some important roles. The technology has been extensively used by the military for many years, so it is not really hard to find inspirations and equipment to develop new valid applications.

Because many of the people involved in the project are working on software development, I found interesting to engage with some hardware stuff. The lab where we are developing the project did not have a hardware-dedicated workplace, so one of the tasks was to make one, which would hugely help to this project development.

2.3 Icarus UAS Platform project

The Icarus UAS Platform project was born in order to study, develop and build an entire multi-purpose airborne system, mounted in an Unmanned Aerial Vehicle. It is being developed by the ICARUS team, a team born in 2006 and comprised by about 20 people working together, both teachers and students from the EPSC.

The Icarus UAS Platform system contains mission and control modules but also payload, which comprises cameras, camcorders (visible and infrared spectrum), sensor acquisition… This makes Icarus UAS Platform project to be multi-purposed, with applications going from fire awareness to surveillance.

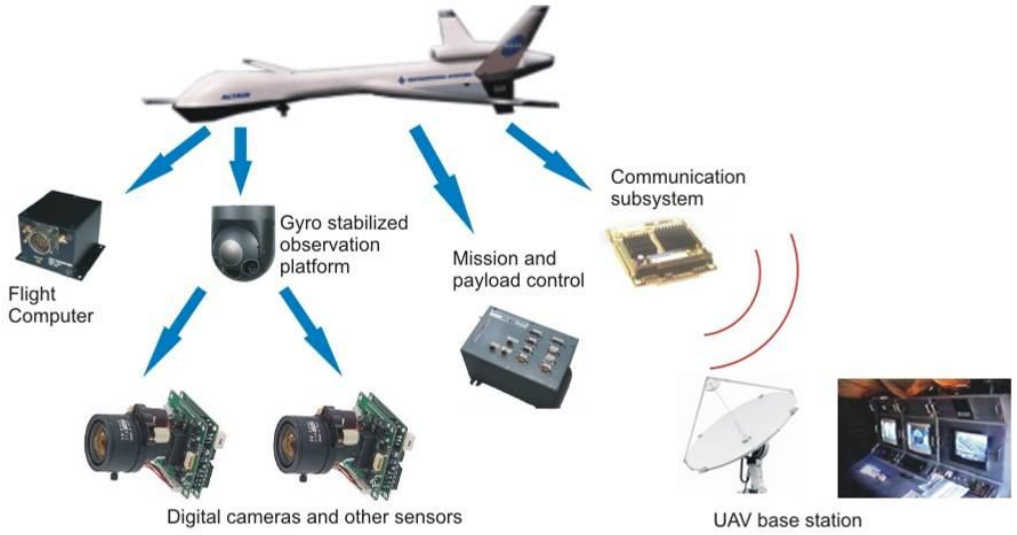


Fig. 2.1 UAS components.

Icarus UAS Platform program will permit to have one or more UAV, connected and interacting between them and operating around ground based stations with a central application in one of it that remotely controls them.

2.4 Technology

2.4.1 Unmanned Aerial Vehicle

Unmanned Aerial Vehicles, UAV in short, are defined as engine powered vehicles that can take off, keep in flight and land with no onboard crew. They can either fly remotely (a ground pilot controls the UAV directly using some kind of remote control) as well as autonomously (a flight path is introduced into the UAV and it just follows it). They are usually called UAS –Unmanned Aircraft System, because the system does not only involve a lonely aircraft but ground systems.

UAS can carry multiple payloads: cameras, sensors, communications equipment and others. They were first used in military tasks, such as for terrain reconnaissance or to attack all kinds of targets (these were called UCAVs, Unmanned Combat Aerial Vehicles). For example, the first known proposal of an UCAV, although it had never been widely used, was an aerostatic globe which had a mechanism that dropped an explosive load after a time-delayed fuse was burnt –it obviously had a very low accuracy. That was on the late 1800s. From then, UCAVs were enhanced, being used as drones for training World War II pilots. During the Cold War they were extensively used for reconnaissance duties. Many different crafts, including VTOL[1] prototypes, were invented, tested and some of them were put in action.

Nowadays, UCAVs are extensively used by armies. Few countries are developing their own autonomous crafts, being the United States, the United Kingdom, Israel and Pakistan at the forefront.

But all these are set within military background. Nevertheless, using them in some kind of civil setting has come not long. Several models for multiple purposes have been developed until today: reconnaissance, mapping, weather and pollution sensing, environment supervision, search and rescue aiding, firefighting, mail delivering… whatever imagination could consider. But there is still a lot of work to do.

Experimental platform: UAV Shadow MK.1

As told, the EFMS will be mounted in an UAV Shadow MK.1, from the Pakistani UAS maker Integrated Dynamics[1]. The EPSC owns two MK.1

The Shadow MK.1 structure is made by Integrated Dynamics, and is a medium-sized UAV. It is based on a classical twin-boom with engine in ‘pusher’ position. Its stock engine is a 250cc boxer featuring about 22CV, which allows a maximum take-off weight (MTOW) of about 90kg.

Fig. 2.2 Integrated Dynamics UAV Shadow MK.1

[1] VTOL: Vertical Take Off and Landing. An aircraft capable of taking off and landing vertically, without needing a runway, yet having the same benefits as an ordinary aircraft has, once in air.

The most likely application for MK.1 will be real-time detection, control and analysis of forest fires. Another possible application for it will probably be the calibration and supervision of VOR stations[2].

2.5 Project justification

Because of the expertise and knowledge obtained by years of expertise in UAS technology by military, some systems are more or less developed, so technology is relatively available from some civil providers.

Few enterprises, such as UAV Navigation[2], provide with very generic autopilot and engine monitoring solutions. However, these systems do not address the mission and payload control, and are not flexible enough to allow the user to easily include these features. This enforces the user to develop its own mission and payload control functionalities, such as an engine and fuel monitor.

The EFMS was somehow theoretically planned before the beginning date of this project. It ranges from the sensors to the Flight Monitor[3] presentation. Data goes through a proprietary service-oriented middleware under development, known as MAREA.

2.5.1 Project objectives

The main aim of this project is to provide a completely new and genuine Engine Management Unit (EMU), ready to be easily implemented next to other systems in an UAS.

The EMU is pretended to be mounted and tested onboard the UAV Shadow, though it must use an open-source, easy to access and understand program code and data output.

The secondary objective of the project is to reprogram the existent engine-dedicated MAREA services in order to interpret the EMU data and publish it into the Flight Monitor screen. The whole system (EMU+MAREA implementation) is the so-called EFMS.

Also, because of the non-existence of an electronics-dedicated workspace in the lab where the EFMS project will be developed, an electronic-specialized workplace is going to be built up. It will include a specific desk, a computer, electronic stuff and tool cabinets and a professional soldering station.

---

[2] VOR station: short of VHF Omni-directional Radio Range is one of the most used radio navigation aid for aircrafts and provides directional and range signal.
[3] Flight Monitor: This is the name of the Icarus UAS Platform flight control interface.

2.6 Project timing

The EFMS development will be distributed in different stages. The hardware part (the EMU part) requires a previous market study, in order to find the desired sensors and then designing the corresponding acquisition circuits. Meanwhile, after that, the entire circuits have to be designed, without forgetting the initial requirements, such as the microcontroller built-in peripherals requirements, the desired outputs and future expansion ports availability. Then, using these electronic schematics, PCB must be developed and manufactured.

At this time, when electronic schematics are more or less done, software development will be started. Software development is also divided in two parts: microcontroller (MCU) program and MAREA corresponding service program. At the beginning, MCU part will be tested using a development board connected to a PC, but as soon as PCBs are available and soldered, program functionalities will be adapted and tested with the circuits, until a 100% working and stable programs are obtained.

Next table (**Table 2.1**) shows this distribution in time:

Table 2.1 EFMS project distribution in time.

Issue		Time
Hardware architecture definition	Documentation	1 week
Identification, buy and import the sensors required for engine and fuel monitoring		2 months
Identification of the acquisition devices		
Design of a microprocessor-based architecture for data acquisition, storage and transmission		3 months
Implement friendly protocol for data communication		
Build a real prototype, implementation within MAREA and working demonstration.		3 months

2.7 Project schematic

Next figure (Fig. 2.3) shows a very simplified block diagram about what the EFMS comprises:

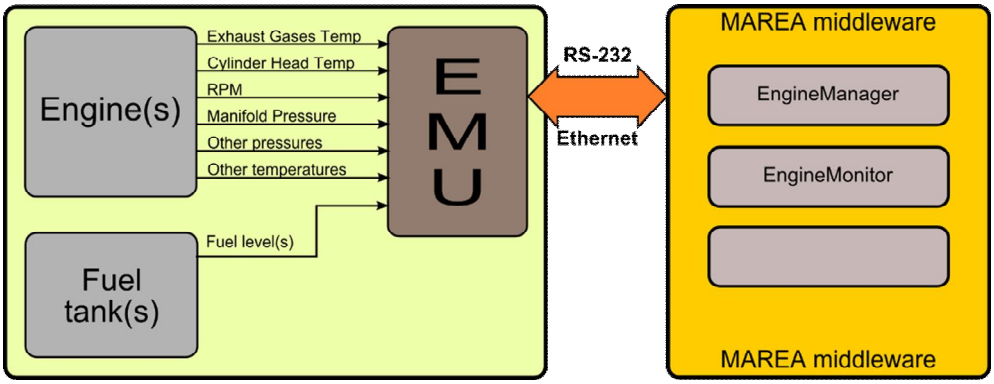


Fig. 2.3. EFMS block diagram.

It is a typical multi-channel measurement system, with parallel acquisition and digitalization circuits for each sensor and a single processing unit. After processing is made, data output is made via RS-232 and Ethernet modules, ready to be read by a computer system.

In this case, MAREA middleware will be adapted to be capable of obtaining this data output stream, so sensing results can be seen in the Flight Monitor screen. Further information on MAREA can be found in “SECTION 6. SOFTWARE DEVELOPMENT”.

SECTION 3. PROBLEM ANALYSIS

3.1.1 Embedded systems

Embedded systems are special-purpose computerized systems. They make the difference with general-purpose computers in that they are designed to do specific-purpose tasks rather than allowing multiple tasks to be ran, as general-purpose computers do. Some of them will run real-time programs, whenever necessary, but other may execute low-performance tasks, allowing them to be simplified and cheapened.

Because of its dedicated task, its design can be optimized. This reduces the complexity and thus the cost of them, allowing its mass production. They usually form part of more complex systems, as if they were subsystems.

Examples of embedded applications are found in calculators, GPS, watches, mobile phones, PDA, videogame consoles, electronic kitchen appliance, and etcetera. These systems run a unique system which only permits limited and specialized operations.

Next figure (Fig. 3.1) shows a typical embedded system distribution:

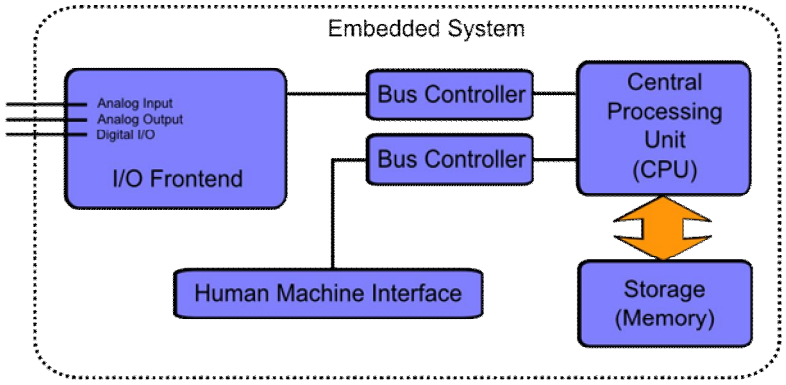


Fig. 3.1. Components of a typical embedded system

The core of these systems is usually a microcontroller (MCU) device. Microcontrollers make the difference with microprocessors in that they integrate several peripherals into a single chip, instead of distributing them in other spaces thus losing physical space and incrementing the overall cost.

This high-integration refers, but not only, to:

- A Central Processing Unit (CPU).
- Volatile memory (RAM) for data storage.
- ROM, EPROM, EEPROM or FLASH memory for program storage.
- General Purpose I/O ports (GPIO), allowing external control or detection.
- Serial I/O ports, featuring UART, SPI, I2C, USB and others.
- ADC, DAC and Comparators.

- Parallel ports.
- Timers, real-time clocks and calendars.
- Self-test and self-control methods.

Microcontrollers are slower at signal processing operations when compared to other similar-purpose devices, such as Digital Signal Processors (DSPs).

However, MCUs are made to do many other types of operations on data. MCUs were designed to provide a huge amount of flexibility through programmability, but at expense of performance. Actually, a DSP can run the same operations as a MCU, but will not have the peripheral flexibility a MCU has. The applications run by a MCU usually have some kind of feedback and carry out several other diverse tasks too. A DSP will likely run a repetitive numeric task, like digital filtering. DSPs are very efficient at repetitive, numerically-intense tasks.

However, to enable handling of high-demanding engineering projects that required both the use of a MCU and a DSP (Digital Signal Processor) in the same circuit, a mutually on-chip combination was invented, known as DSC (Digital Signal Controller).

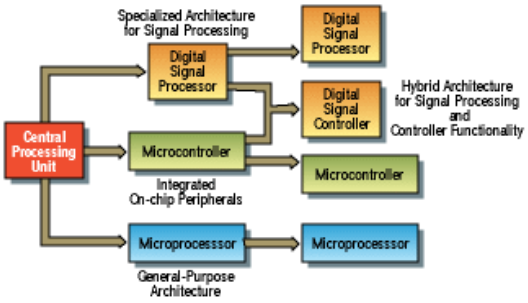


Fig. 3.2 The DSC is a combination of both MCU and DSP.

3.1.2 Digital Signal Controller

DSCs are hybrid systems, combining typical microcontroller functionality along with a DSP processor. The DSC term was adopted by Microchip Technology[3] in 2002, when they introduced their 6000 series of DSCs. The DSC term was going to be adopted by most of DSCs manufacturers. Even so, Microchip was going to stay at the forefront of MCU and DSC manufacturing, along with Texas Instruments[4] and Freescale[5], being the #1 seller of 8-bit MCUs since 2002.

Curiously on February 2008, Microchip announced the expedition of its six billionth PIC microcontroller.

PIC is the acronym of Programmable Intelligent Computer, and is the prefix used by Microchip Technology to name its microcontrollers. PICs are so popular because of their low-cost, characteristics, wide-ranging availability, easy programming and re-programming, and because of the free development tools (although some are not free, they are far cheaper than other manufacturers).

3.2 Microchip and dsPIC

Microchip PICs range from 8-bit to 32-bit. For making the difference between traditional PICs and DSP-powered PICs, Microchip distinguishes between PIC and dsPIC.

The figure below shows the entire Microchip PIC family.

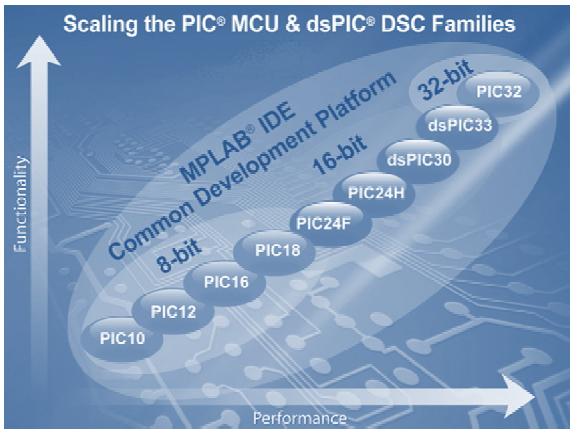


Fig. 3.3 The Microchip Technology PIC® microprocessors family.

Microchip devices feature these general advantages in their web page:

- Powerful architecture.
- Range of memory technologies: Self Programming Flash, OTP, ROM.
- Complete portfolio: 6 to 100 pins, 384B to 512KB of program memory, up to 80MHz.
- Comprehensive and affordable (sometimes free), easy to use development tools: compilers, assemblers, programmers, simulators, in-circuit emulators.
- Complete technical documentation and post design-in support
- Low-risk product development & faster time to market.
- Lower total system cost.
- Easy migration across product families:
- Upward compatible architectures to preserve investment in code development.
- Pin compatibility in multiple packages facilitates drop-in replacement.
- Easy Migration across 8, 16, and 32-bit families.

3.2.1 Why a dsPIC33?

Microchip devices feature the best overall relationship between quality and price, and also grant access to a wide availability and range of development tools and application notes via Internet.

It had been kept in mind since the beginning that a powerful device was required. Although EFMS is not an exhaustive resource consumer, both future hardware expansion and software modifications will probably be. For instance,

one future system that will probably be programmed within the same dsPIC along with the EFMS will be an Electrical Management Service, called ELMS.

16-bit family

Microchip features four families of products within the 16-bit range. These include both MCUs and DSCs. Common attributes among all of them are:

- Pinout compatibility.
- Software compatibility.
- Peripheral compatibility
- Common development tools.

Fig. 3.4 shows a comparison chart between these families.

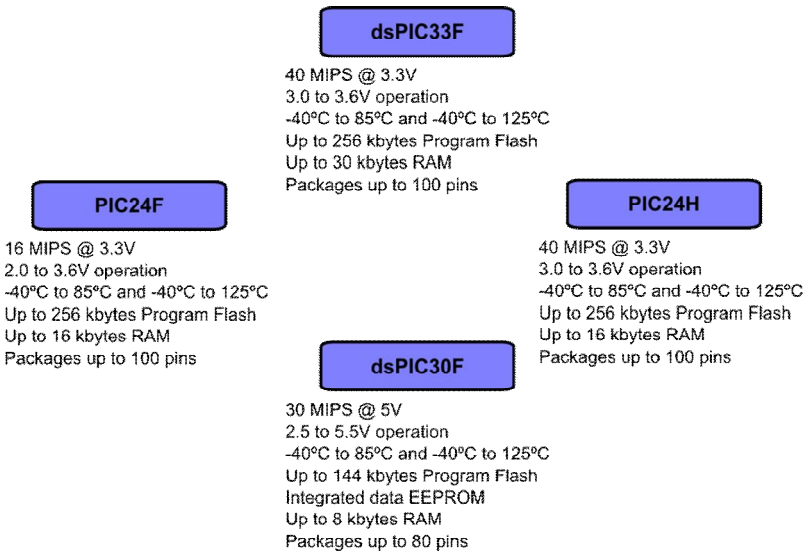


Fig. 3.4 Microchip 16-bit families of products

An advanced device such as the dsPIC33FJ256FP710, with a lot of available calculus power as well as program and data memory, was the most logical PIC to be used. Meanwhile, it was unnecessary for the applications to use a 32-bit-long data MCU, so a 16-bit DSC featuring DSP power was a better choice.

3.2.2 Development tools

One of the toughest points of Microchip is that some of their development tools are free to download on the Internet. Others, such as the C compiler, are relatively cheaper when compared to other MCU providers. Some of these tools have been used in this project, and are rapidly analyzed below.

Microchip MPLAB IDE

MPLAB Integrated Development Environment (IDE) is a free, integrated toolset for the development of embedded applications employing Microchip's PIC and dsPIC microcontrollers.

MPLAB IDE includes a host of free software components for fast application development and debugging. MPLAB IDE also serves as a single, unified graphical user interface for additional Microchip and third party software and hardware development tools. Moving between tools is a snap, and upgrading from the free built-in software simulator to hardware debug and programming tools is done in a flash because MPLAB IDE has the same user interface for all tools.

Microchip C compiler (C30) is free to download, and will work unlimitedly. Its only restriction is that after a period of time (60 days) the code optimizations cannot be used anymore.

Microchip Explorer 16 development board

The Explorer 16 development board is intended to evaluate the features and performance of Microchip's PIC24 Microcontroller, the dsPIC33 Digital Signal Controller (DSC) families, and the new 32-bit PIC32MX devices.

Coupled with the MPLAB ICD2 In-Circuit Debugger or MPLAB REAL ICE, real-time emulation and debug facilities, it speeds up the evaluation and prototyping of application circuitry.

Microchip ICD2

ICD2 main function is real-time debugging of applications, but it can also serve as a programmer. With the In-Circuit debug functions, programs can be loaded, executed in real-time and examined in detail with MPLAB debug functions. Programs can be executed step-by-step, and *watch* variables can be set so close monitoring of program progression can be done.

Its only requirements are three reserved PIC device pins: MCLR, RB6 and RB7.

Altium Designer

Altium Designer comes to be the next generation of electronics design programs. It is the successor to the successful P-CAD 2006.

One of the best features is its unique integration in a single stand-alone program of schematic and PCB design, project development and simulation. Even though it is a little difficult to become friendly with its huge amount of functions, it is a powerful software which allows the user to make whatever kind of electronic design, in both hardware and software stages.

SECTION 4. GENERAL SPECIFICATIONS

4.1 General architecture

The EFMS system will be implemented next to other systems that are already included into the UAV, and will communicate with the middleware using the existent Ethernet line.

An RS-232 port is also included in the module, which intend is twofold: permit a direct connection to the module when the plane is on the ground, but also to allow auxiliary communication in case of failure in MAREA net. Initially and at the presentation date, the access to EMU data is only possible via the RS-232 port.

Hence, the general architecture is shown in Fig. 4.1:

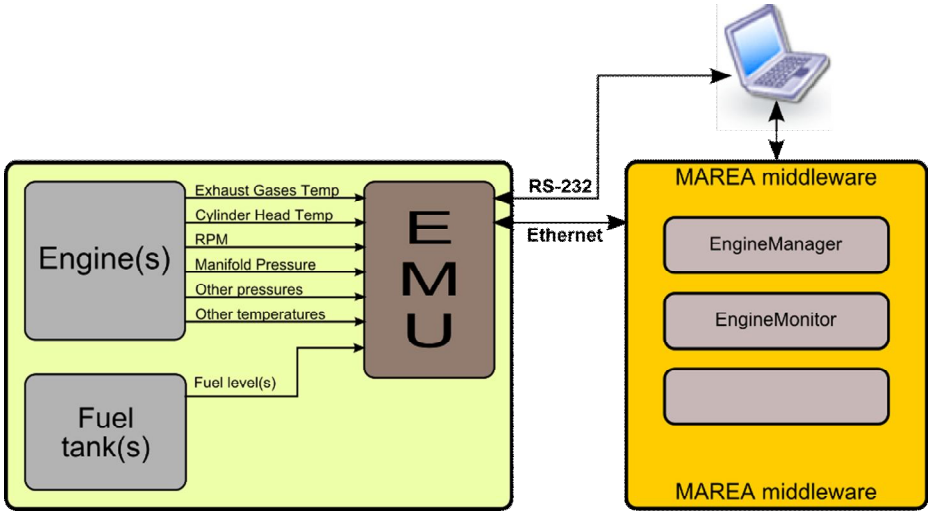


Fig. 4.1 EFMS general architecture

The EMU keeps monitoring all data continuously, at a refresh rate of about 300 milliseconds, and stores it inside known memory registers, which can eventually be accessed by the Middleware service (in this case, MAREA *EngineManager* service) to obtain this newest data and then showing it in the Flight Monitor screen (in MAREA, *EngineMonitor* service).

4.2 Electronic scheme

Electronically speaking, EMU layout is of a typical multi-channel, with one of the SPI buses (SPI1) interconnecting the different sensing devices. This SPI bus also has two expansion ports, each one allowing up to three more devices to be connected to them, for future purposes.

Each module has its own power conditioning stage, fed by 12V available from the on-board batteries. The Modules communicate between them via the SPI1 bus.

As outputs, Ethernet and RS-232 connections are available. Ethernet module has a dedicated SPI bus (SPI2), which will permit full-time access. ICD2 connection is dedicated to DSC programming and status-monitoring purposes.

An overview to this electronic scheme can be seen in Fig. 4.2.

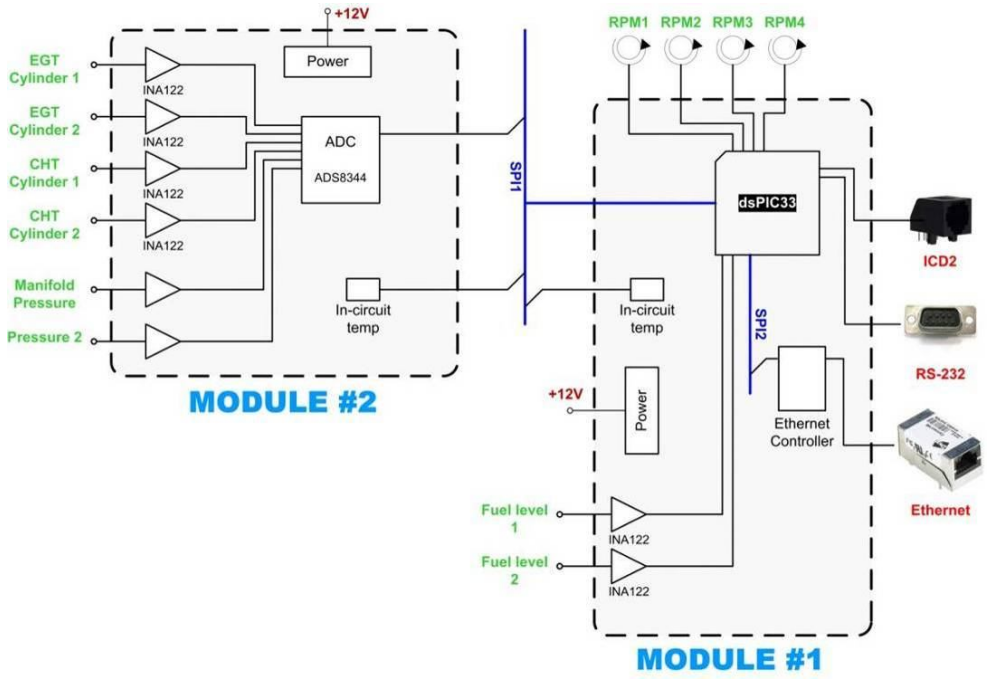


Fig. 4.2 Basic electronic scheme

4.3 *EngineManager* and *EngineMonitor* services

EngineManager is the name of the MAREA service that interfaces with EMU. It has the responsibility of creating a new RS-232 connection (at the time of writing this documentation, only RS-232 connection is functional. Ethernet module is present, but not working yet. It will be implemented after the project presentation date).

It also has to request the corresponding the EMU data, store it into dedicated public registers and also to maintain a correct connection with it.

EngineMonitor service is the name of the service which takes the data from these known registers, then processes them and publishes them into the Flight Monitor. It also has to control the early-alarm systems, and allow some kind of alarms configuration (alarm priority, units, ranges…).

A deep analysis to these services is done in SECTION 6. SOFTWARE DEVELOPMENT.

SECTION 5. HARDWARE DEVELOPMENT

5.1 Specifications

As told, the main purpose is to provide a generic system, with a fixed number of sensing inputs but with two additional expansion connectors are included, to allow further inclusion of any of sensors.

The default built-in sensing options are:

- Fuel related:
  - Two fuel tanks level.
- Engine related:
  - Two Cylinder Head Temperature (CHT).
  - Two Exhaust Gases Temperature (EGT).
  - Two pressure sensors (e.g. manifold pressure).
  - Four hall-effect sensors (e.g. engine RPM).
- Other:
  - Circuitry temperature.
  - Two 3-channel SPI expansion connectors.

5.2 Hardware modules location

Because sensors are distributed by two locations inside the UAV, two separate circuitry modules are used, just to avoid sensor errors and EMI-related problems but also to save wire.

These two modules are called Module #1 and Module #2.

Basically, Module #1 plays the main role, as it contains all signal processing parts, sensor connections and other I/O connections. Meanwhile, Module #2 acquires, filters and then converts these incoming high-noise signals from the engine (CHT and EGT thermocouple signals), and also the manifold pressure readings.

Fig. 5.1 shows how these two modules will be probably distributed along UAV Shadow.

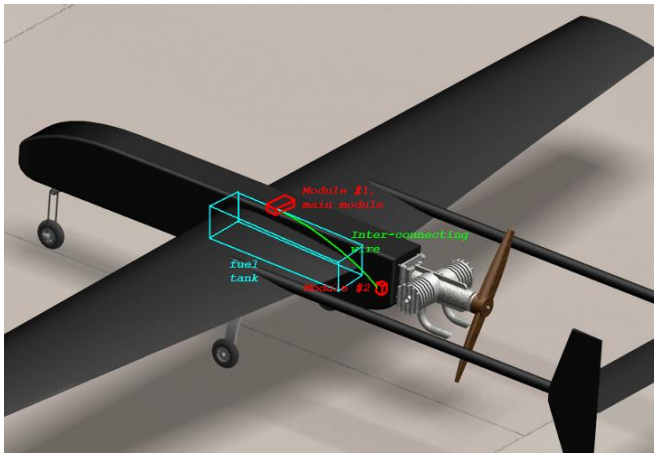


Fig. 5.1 Possible EFMS Modules distribution along UAV Shadow.

5.2.1 EFMS Module #1

Module #1 is placed inside a small 6.5x12x4 ABS, virtually-EMI-protected enclosure. Its small dimensions and robustness allow it to be mounted almost anywhere, preferably far away from noisy elements, so it will be probably mounted at the middle of the Shadow fuselage; just above the fuel tank and distant to the engine (refer to Fig. 5.1).

Because of the small dimensions available inside the enclosure, two printed circuit boards are used, both containing electronic stuff, integrated circuits and connectors.

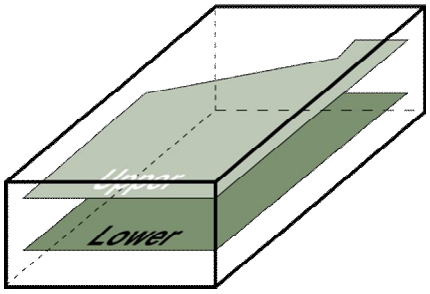


Fig. 5.2 PCB distribution within Module #1.

Lower printed circuit board is four-layered, and contains:

- Top Layer:
  - Main processing unit (dsPIC33FJ256GP710).
  - RS-232 driver.
  - RS-232 connector.
  - ICD2 connector.
  - In-circuit temperature sensor.
  - Analog circuitry and connectors.
- Upper Mid-Layer: 3.3 Power Plane
- Lower Mid-Later: Ground Plane

- Bottom Layer: printed circuit only. No IC, as there is none room between this side of the board and the bottom side of the enclosure.

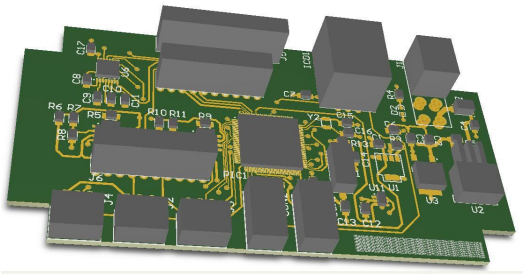


Fig. 5.3 3D sketch of the lower EFMS Module #1 circuit board.

Upper printed circuit board is also four-layered, and contains:

- Top Layer:
  - Ethernet controller.
  - Ethernet connector.
  - Analog circuitry and connectors.
- Upper Mid-Layer: 3.3 Power Plane
- Lower Mid-Later: Ground Plane
- Bottom Layer: printed circuit only.

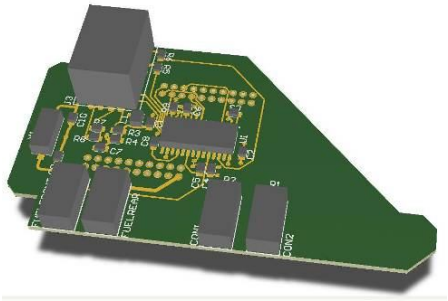


Fig. 5.4 3D sketch of the upper EFMS Module #1 circuit board.

5.2.2 EFMS Module #2

Module #2 is placed inside a 5.5x5.5x4 ABS enclosure, being made of the same material as Module #1. Because Module #2 contains most engine-specific sensing inputs (thermocouples and pressure sensors), it should be located as close as possible to the engine.

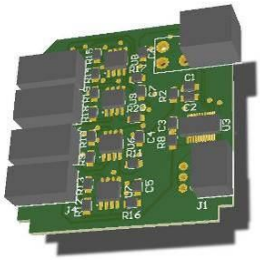


Fig. 5.5 3D sketch of the EFMS Module #2 circuit board.

In UAV Shadow, this module will be located in the air-filter bay, next to the electronic ignition system.

5.3 Sensors: Fuel Level

For controlling the fuel level inside the fuel tanks, these two sensing solutions are the most typically used, not only in aircraft applications, also in industrial environments:

- Floating-element.
- Capacitive.

The difference between them is both the measurement procedure and the output signal. Whilst floating-element uses a ball-cock in contact with fuel and gives a resistive output, capacitive uses the dielectric variation caused by the level change between two plates to measure it.

5.3.1 Shadow's fuel tank physical specifications

UAV Shadow uses a single fuel tank, which is located in the middle part of the fuselage, just between the avionics bay and the engine bay. It is artisanal-made of fiber-glass and can hold up to 25 liters of fuel approximately.

Because of the typical longitudinal movements of an aircraft (ascending and descending), it has internal built-in ribs, which are supposed to avoid drastic fluid movements which could lead to engine failures and even a loss of aircraft control.

Hence, and also because of tank dimensions of UAV Shadow, two sensors are used. They will be located within the longitudinal tank perimeter, as shown in Fig. 5.6.

Fig. 5.6 Fuel level sensors location in UAV Shadow MK.1

5.3.2 ISSPRO fuel sensors and adaptation circuit

The sensors to be used are floating-element, in-tube type. They are made by ISSPRO[8], and are thought to be fitted in OEM automotive fuel tanks, but they are perfectly usable for aeronautical applications. Their output is resistive and linear, ranging from 33Ω (float in upper position, full tank) to 240Ω (float in lower position, no fuel).

Fuel sender adaptation circuit

The output is resistive. The Wheatstone bridge is a widely-used circuit for measuring resistance variations and turning it to a voltage-relative signal, and is shown below in Fig. 5.7.

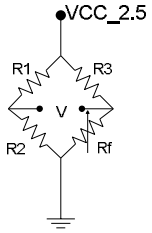


Fig. 5.7 Voltage fed Wheatstone bridge.

$$V = VCC\_2.5 \cdot \left(\frac{Rf}{Rf + R3} - \frac{R2}{R2 + R1} \right)$$

With this procedure, a level (which should be understood as a percentage) dependant voltage, *V*, is obtained. In Wheatstone bridge circuits, stability in resistances and in source becomes necessary. To suit the analog to digital converter requirements, the input voltage must range from 0 to 2.5V. But because of the low resistance of the ISSPRO sensor, it is not permissible to get such a relatively high voltage at the bridge output, because current consumption when the sensor resistance is low would be too high.

This is resolved by having a low-voltage output and then using an instrumentation amplifier, which will also filter and gain the low signal.

A Burr-Brown INA122 (Fig. 5.8) instrumentation amplifier is used for this purpose. It is a precision, low noise, high CMRR[1] differential signal amplifier. Its gain is adjustable from 5 to 1000 via an external resistor.

[1] The common-mode rejection ratio (CMRR) of a differential amplifier measures the tendency of an amplifier to reject input signals which are common to both input leads. A high CMRR is important in applications where the signal of interest is represented by a small voltage fluctuation superimposed on a (possibly large) voltage offset, or when relevant information is contained in the voltage difference between two signals.

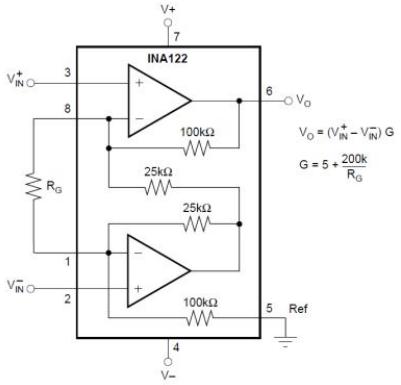


Fig. 5.8 INA122 internal schematic.

The Wheatstone bridge is designed in order to provide a 0 to 0.430V output, making the bridge to consume just 2mA as maximum. The INA122 goes next to the output, taking the differential voltage signal directly into its inputs. Using a 220kΩ resistor between the gain pins sets the gain to 5.9 V/V. This will make the output voltage to range from about 0V to 2.5V, suiting the reference voltage at the analog to digital converter.

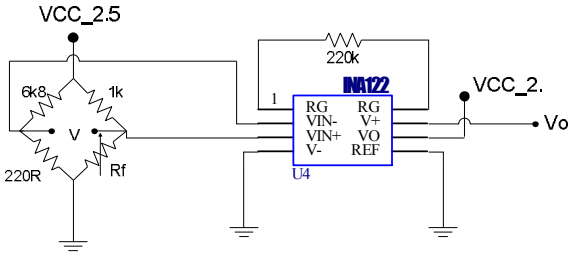


Fig. 5.9 Fuel level sensors adaptation circuit.

A/D Conversion and dsPIC interfacing

The voltage output from the INA122 is then connected to microcontroller AN4 and AN5 pins (one for each sensor). These are 10-bit built-in analog to digital converters, and feature:

- Successive Approximation (SAR) conversion
- Conversion speeds of up to 1.1 MSPS
- Automatic Channel Scan mode
- Selectable conversion trigger source
- Selectable Buffer Fill modes
- Four result alignment options (signed/unsigned, fractional/integer)

It has been preferred to use the 10-bit in-PIC analog to digital converters rather than a dedicated 16-bit device because the low precision available from the ISSPRO sensor, which would have made it worthless.

5.4 Sensors: CHT, EGT and temperature

Engine running temperatures are the most basic parameters to be supervised to ensure its correct and safe operation. Since the air-fuel mix can only be modified with the engine turned off (at least in basic engines like the 240iB2, not big ones), the only way to guarantee that it is running fine is by checking that temperatures ensuring that they keep inside the nominal range.

These temperatures are:

- Cylinder Head Temperature (CHT)
- Exhaust Gas Temperature (EGT)

Checking both CHT and EGT comes to be the best way to find out whether the engine is overheating or excessively cool. These problems appear usually because of an incorrect adjust of the air-fuel ratio when in ground. Using them together could lead to think that they work as redundant sensors, because on final stage they are used to control the same, but really it is not.

Generally, the leaner the air-fuel mixture is, the hotter the engine will run and thus more RPM will achieve. But this has a limit, and if the mix is leaned yet more, the combustion temperature will start to go down. In the real life this effect can be seen in an oxy-acetylene torch inside a room where the amount of oxygen can be adjusted. When the torch is lighted up, the mixture will have an excess of fuel (acetylene) compared to the existent oxygen. The flame will be yellow colored, producing a lot of smoke and not very hot. If more oxygen is gradually introduced, the flame turns bright blue, the smoke disappears and the flame temperature rises. But if more oxygen is introduced beyond that point, the flame will start to shrink, so its temperature will decrease until it disappears. The same happens inside the engine combustion chamber.

Hence, when the mix starts to be lean, CHT will gradually increase, but is not really an indication of a hotter combustion. It is most likely a sign that detonation effect[2] (also known as 'knocking') is happening. Detonation floods the combustion chamber with heat, so CHT goes up, but at contrast, as it happened with the Oxy-Acetylene torch when it had an oxygen glut, EGT will go down. When this occurs, engine will rev-down, but the engine block will be still excessively hot. This could cause permanent damage to the engine if not quickly cooled down.

The quickest way to force down the temperature when the UAV is flying is to decrease its RPM, then bringing the UAV back and readjusting the carburetor settings to a correct setting so permanent engine damage is avoided.

More information on this can be found on [7].

---

2 Detonation effect: Is the collision of two flame fronts inside the combustion chamber, where it actually should only be one, and it is the most frequent cause of heat related engine failures. The unwanted flame front is created because of a combination of high temperature and pressure, which ignites the remaining unburned fuel/air mixture beyond the flame front.

Temperature measurement

Because of the variety of temperature ranges to be monitored, some different kinds of temperature sensors can be used. These are:

- Thermocouples.
- RTD (Resistance Temperature Detectors).
- Silicon integrated circuits.

Table 5.1 is a quick-reference to show the most common differences between them:

Table 5.1 Differences between RTD, thermocouples and silicon IC.

	RTD	Thermocouple	Silicon IC
Temp. Range	-200 to 850°C	-184 to 1260°C	-55 to +125°C
Temp. Accuracy	Class B = $\pm[0.012 + (0.0019 \cdot \lvert t \rvert) - 6 \cdot 10^{-7} t^2]$	Greater of ±2.2°C or ±0.75%	Various, ±0.5 to 3°C
Output signal	≈ 0.00385 Ω/Ω/°C	Voltage (40 µ/°C)	Analog, Serial, Logic, Duty Cycle
Linearity	Excellent	Fair	Good
Precision	Excellent	Fair	Fair
Durability	Good, Wire wound prone to open-circuit vibration failures	Good at lower temps. Poor at high temps. Open-circuit vibration failures	Excellent
Thermal response time	Fast (function of probe material)	Fast (function of probe material)	Slow
Cost	Wire wound - High, Thin film - Moderate	Low	Moderate
Package options	Many	Many	Limited, IC packages
Interface options	Small ΔR/Δt	Cold junction compensation, Small ΔV	Sensor is located on PCB

5.4.1 Thermocouples: CHT and EGT

Thermocouples are the most common sensors used in high-temperature, low-precision measurements. They are based in Seebeck, Peltier and Thomson effects, which, in short, explain how an electromagnetic force results when different metals are joined together and that is temperature-dependant too.

Its electronic symbol is shown in Fig. 5.10:

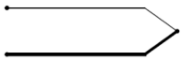


Fig. 5.10 Thermocouple symbol and appearance.

A thermocouple consists of two wires of dissimilar metals that are soldered together at one end as shown in Fig. 5.11. The temperature at the Reference Junction (also known as the Cold Junction Compensation Point) is used to negate the errors contributed by the Iron-Copper and Constantan-Copper junctions.

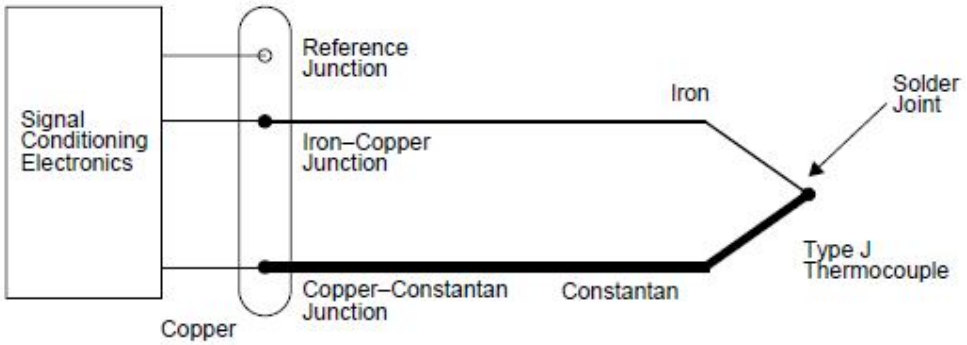


Fig. 5.11 Thermocouple system parts.

The solder point of the thermocouple is positioned on the target where the temperature measurement is needed.

This configuration of materials produces a voltage between the two wires at the unsoldered end that is a function of the temperature of all of the junctions. Hence, the thermocouple does not require voltage or current excitation.

Since a voltage appears at the open end of the two dissimilar wires, it would seem as if the thermocouple interface could be done in a straight forward manner just by measuring the voltage difference between the wires. This would be acceptable if it was not for the fact that the termination ends of the thermocouple connect to another metal, usually wire copper or a connector. This creates another pair of thermocouples, which introduces a significant error to the system. The only way to negate this error is to sense the temperature at the Reference Junction box (Fig. 5.11) and subtract the contributing errors of these connections in a hardware solution or a combination of software and hardware.

Pure hardware calibration techniques are more limited in terms of linearization correction rather than a software and hardware combination techniques. Usually a RTD, a thermistor or an integrated silicon sensor are used to sense this junction temperature accurately.

The thermocouple effect can be found from in every combination of two dissimilar metals; however, standard combinations of these two metals have been regularized because of their desirable linearity and their voltage magnitude drop versus temperature. These common thermocouple types are called E, J, T, K, N, S, B, and R (**Table 5.2**).

Table 5.2 Common thermocouple types.

Type	Conductors	Conductors color		Temperature Range (ºC)	Seebeck Coefficient	Application Environments
		Thermocouple Grade	Extension Grade			
E	Chromel & Constantan			-200 to 900	60mV/°C	oxidizing, inert, vacuum
J	Iron & Constantan			0 to 760	51mV/°C	vacuum, oxidizing reducing, inert

Type	Conductors	Conductors color		Temperature Range (ºC)	Seebeck Coefficient	Application Environments
		Thermocouple Grade	Extension Grade			
T	Copper & Constantan			-200 to 371	40mV/°C	corrosive, moist, subzero
K	Chromel & Alumel			-200 to 1260	40mV/°C	completely inert
N	Nicrosil & Nisil			0 to 1260	38mV/°C	oxidizing
S	Platinum (10% Rhodium) & Platinum	NONE ESTABLISHED		0 to 1480	11mV/°C	oxidizing, inert
B	Platinum (30% Rhodium) & Platinum (6% Rhodium)	NONE ESTABLISHED		0 to 1820	8mV/°C	oxidizing, inert
R	Platinum (13% Rhodium) & Platinum	NONE ESTABLISHED		0 to 1480	12mV/°C	oxidizing, inert

Thermocouples are highly non-linear when compared to RTD, thermistor and integrated silicon sensors. Consequently, complex algorithms must be performed at the processor, or look-up tables might be used.

Application within EFMS

In EFMS, thermocouples are used to measure CHT and EGT. In this project, CHT probe is made by VDO whilst EGT probe is made by Westach[8]. Nonetheless, CHT probe must be of J-type, while EGT must be Type K.

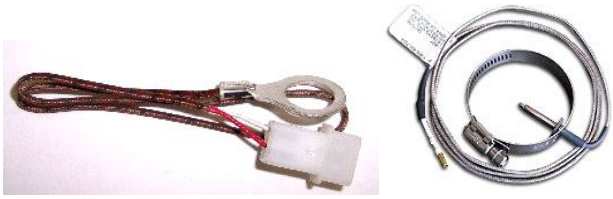


Fig. 5.12 CHT probe (left) and EGT probe (right)

Acquisition circuit: Module #2

CHT and EGT thermocouples are connected directly to Module #2, which is located in the engine filter air bay; this is, at about 40cm from the solder point.

Because of the noisy environment surrounding the thermocouple wires when they go away from the testing point towards the connector in Module #2, high CMRR, low-noise instrumentation amplifiers are used in the acquisition part inside the module. Then the output signal is converted through a 16-bit ADC and transmitted via SPI bus to the Module #1 (refer to Fig. 5.1).

Again, a Burr-Brown INA122 instrumentation amplifier is used for this purpose. Remember that this is a precision, low noise, high CMRR differential signal amplifier. Its internal schematic is shown again in Fig. 5.13.

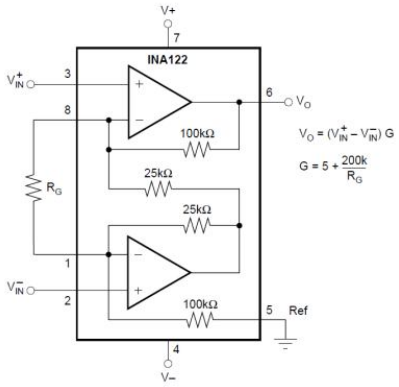


Fig. 5.13 INA122 internal schematic.

This time, the amplifier gain varies if it is for CHT to EGT signal amplification, because of the different metal combination used in each thermocouple. While CHT uses a Type J thermocouple, EGT uses a Type K thermocouple.

The temperature at the reference joint (located just in the connection point between the thermocouple wires and the PCB connector) is also measured with an integrated silicon sensor (TMP121), which will be detailed in section 5.4.3.

A schematic showing only one of the Type J thermocouple acquisition circuits is shown below in Fig. 5.14.

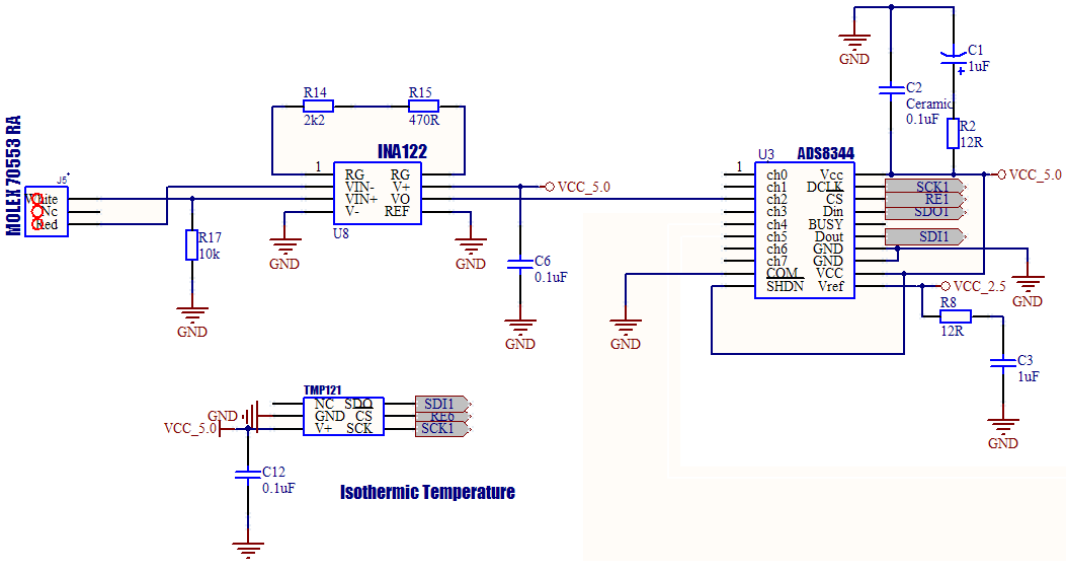


Fig. 5.14 A single thermocouple acquisition circuit.

This circuit is located inside Module #2. The ADC has more circuits connected to its input channels, but only one is shown because of readability reasons (see "**Table 5.3** Module #2 ADC inputs review and selection.").

Note also that the ADC and the Isothermal Temperature circuit are connected to the same SPI bus lines. For more information regarding the ADS8344, see "5.7 External analog to digital conversion".

5.4.2 Resistance Temperature Detectors

Application within EFMS

At the beginning, RTDs were thought to be used as the only temperature measurement option, because of its precision and full-scale range. But after better studying the physical characteristics of its positioning inside the UAV elements, other options where then preferred, such as OEM thermocouples for engine measurements, or integrated circuits for PCB measurements.

So finally no RTDs have been used. This does not mean that a future implementation of them is not possible, because this shall be easily made trough the existent expansion bay ports.

Information on RTDs can be found on "Anexos" documentation.

5.4.3 Silicon Integrated Circuits

The integrated circuit temperature sensors offer a good alternative for non-harsh environments. They are built in a known format package, so they are very easy to install in PCBs. They usually include extensive signal processing circuitry, providing a user-friendly output, such as linear analog voltage or current signal, or digital SPI, I2C, etcetera. As a con, range is not extensive on these sensors. In most cases, this range does not go far behind -55ºC and 125ºC.

Applications

Common applications for these sensors are precision, low-ranged temperatures such as thermocouples cold junction compensation, circuitry temperature in personal computers, office electronics and cellular phones, and actual temperature on HVAC systems.

Application within EFMS

Two silicon integrated circuits will be used, one on each Module. Module #1 sensor will measure in-circuit temperature, while Module #2 will measure the cold junction temperature present in the connection between thermocouple wires and PCB connector.

The sensor to be used is designated TMP121, from Texas Instruments. It features SPI output, 1.5ºC accuracy and -40ºC to 125ºC measurement range. Because of its SPI-ready output, it is very easy to implement along with other elements connected to the primary SPI bus (SPI1) of the DSC.

Next figure (Fig. 5.15) show its external appearance and pins distribution:

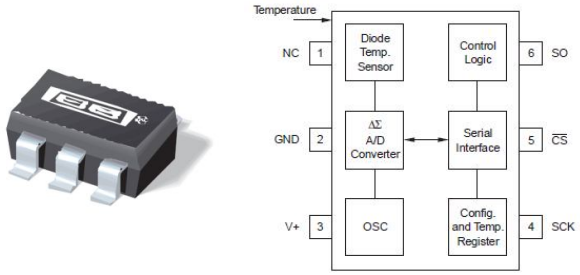


Fig. 5.15 Texas Instruments (Burr-Brown) TMP121 temperature sensor.

The TMP121 device use is very simple. With power and `CS` high, it keeps sampling and converting the temperature. When `CS` turns low, conversion stops and digital data is latched into the output register, prior to being clocked into the first falling `SCK` edge.

As told, two TMP121 are used along with the EFMS. Each SPI lines are connected to SPI1 bus, and `CS` lines are connected to `RE0` in Module #1 IC, and `RE6` in Module #2 IC. Generic connection schematic is shown in the figure below (Fig. 5.16). The capacitor acts as a transient suppression device.

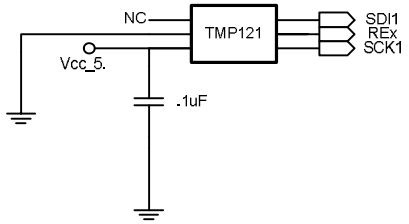


Fig. 5.16 TMP121 circuit connection schematic.

5.5 Sensors: RPM from engine and wheels

RPM measurement is made from Hall-Effect sensors.

Hall-effect was discovered in 1879 by Edwin Herbert Hall while working on his PhD thesis in John Hopkins University, Baltimore. In short, Hall discovered that if an electrical conductor is put among a magnetic field, a potential difference is created (which is perpendicular to both of them).

Fig. 5.17 shows this effect. With no magnetic field present, current distribution along the plate is uniform and there is no potential difference in its terminals.

But if a magnetic field with *B* density is present, the electrons are excited by a Lorentz force[3] creating a potential difference (Hall voltage V_H), which is proportional to the electrical current and the magnetic field, but inversely proportional to the plate thickness.

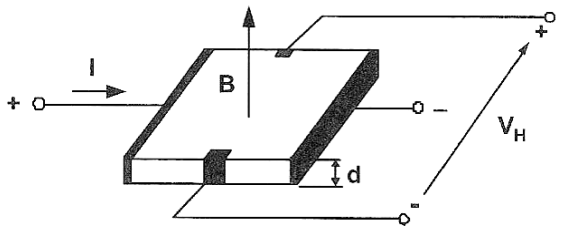


Fig. 5.17 Hall Effect basis.

Hall-Effect sensors have lots of purposes, because of its unique features:

- Solid state devices.
- Long life.
- Relatively high working frequency (more than 100kHz).
- No moving parts.
- High compatibility, especially in digital output devices.
- High working temperature range.
- Good repeatability.

Depending on its output, Hall-Effect sensors can be analog (linear voltage or current output) or digital. Digital ones make the difference with analog in that they have a Schmitt-Trigger before the output transistor (Fig. 5.18).

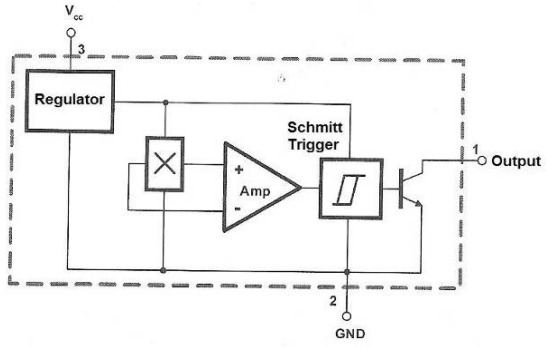


Fig. 5.18 Block diagram of a digital Hall-Effect sensor.

Because of the output transistor, which works as open-collector, a pull-up resistor is needed at the output to guarantee a high-level signal when the transistor is short-circuited to ground.

[3] Lorentz force: In physics, the Lorentz force is the force on a point charge due to electromagnetic fields.

Application within EFMS

Hall-Effect sensors are used mainly, but not only, to measure engine RPM. Four RPM inputs are available in EFMS.

The sensors are from Hamlin, product name 55100-3M-02-A (Fig. 5.19).

Fig. 5.19 Hamlin 55100-3M-02-A, Hall-Effect sensor.

It is a 3-wire (open collector output), featuring switching speeds up to 10kHz, digital voltage output, long life and an easy to mount shape. It will be fed with 3.3V, and signal wire will be pulled up to it through a 6.8kΩ resistor.

Signal output will be connected to the DSC Input Capture module.

5.5.1 Input Capture module

The Input Capture module is the ideal peripheral to calculate RPM.

This module uses an external input signal to capture timer values. For example, it can be used to store two different timer values corresponding to two consecutive input signals, subtract them and invert them in order to obtain its frequency of repetition.

The dsPIC33FJ256GP710 offers up to four of these modules, in pins RD8 to RD11. It can be set-up to capture both signal rising and/or falling edges, and cause an interruption every first, second, third or fourth capture event. All of the Input Capture modules in the microcontroller are used, as one of the requisites was to have four RPM calculation modules.

5.6 Sensors: Absolute pressure measurement

Absolute pressure measurement quantifies the pressure value in contrast with the vacuum. For example, typical atmospheric pressure readings are done with vacuum as a reference.

Other pressure measurements can be differential (pressure difference between two different points), or relative (similar to differential, but one of the points is at actual atmospheric pressure).

Many of the existent pressure sensors use some kind of bendable diaphragm mechanism to measure it (see Fig. 5.20). This deformable element directly

actuates over an electronic sensor. This sensor determines the output of the pressure sensor, and can be resistive, inductive, capacitive, piezoelectric…

Silicon integrated pressure measurement circuits can also be found. One of their advantages of them is that they are usually temperature-compensated and calibrated, so its precision is higher. The output is usually in a linear analog voltage form, proportional to the applied pressure.

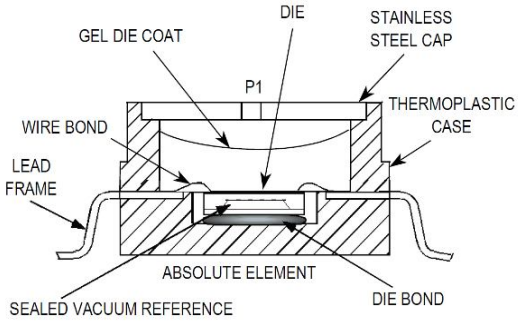


Fig. 5.20 MPX4100A absolute pressure sensor cross section.

Application within EFMS

EFMS mounts two absolute pressure sensors, both located inside Module #2. These sensors are from Freescale, product MPX4100A, and they are especially designed for manifold pressure measurement in engines. Its appearance can be seen in Fig. 5.21:

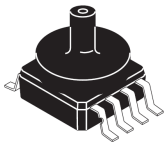


Fig. 5.21 Freescale MPXA

It can measure up to 15.2 PSI (105kPa), giving a linear output voltage from 0 to 5V (see Fig. 5.22).

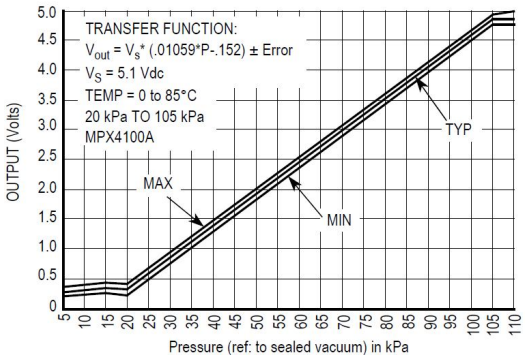


Fig. 5.22 MPX4100A absolute pressure sensor transfer graph.

Acquisition circuit

As told, both MPX4100A are mounted in Module #2, next to thermocouple probes acquisition circuits. They are connected to the same ADC as these thermocouples, in channels 0 and 1 (see “5.7 External analog to digital conversion” for more information on the ADC).

Because they give a voltage output that ranges from about 0.3V to 5V, and the ADC voltage reference is at 2.5V, a ½ gain attenuator circuit is needed between the sensor and the converter. A simple general-purpose non-inverter operational amplifier (Texas Instruments TL052) is used to this purpose (Fig. 5.23).

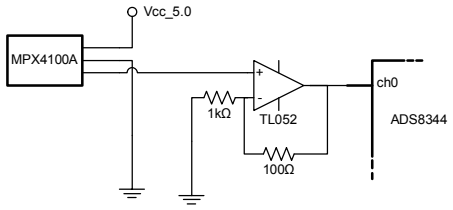


Fig. 5.23 Absolute pressure sensor adaptation circuit.

5.7 External analog to digital conversion

For these signals coming from a voltage-output adaptation circuit, an analog to digital conversion must be made in order to send the corresponding digital signal to the microcontroller.

Temperatures from cylinders (CHT), from exhausts (EGT) and pressure measurements are processed and digitalized through circuitry containing an ADS8344 at the ‘end’.

ADS8344 is 16-bit, 8-channel, SAR and SPI interfaced. It is sourced from 3.3 to 5V, although as low as 2.7V is allowed. The figure below (Fig. 5.24) shows an internal ADS8344 block schematic.

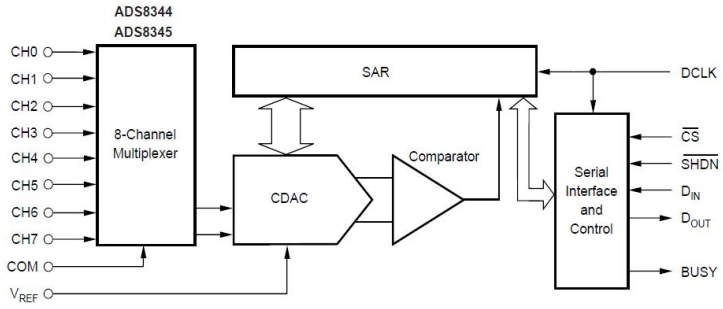


Fig. 5.24 ADS8344 internal block diagram.

The ADS8344 is connected via SPI bus to SPI1 bus present in the whole EFMS system. It is connected in slave mode. This means that it receives its clock

signal from an external source, which is the microcontroller in this case. This signal shifts data in and out of the device, and also controls the conversion.

It is microcontroller program duty to send the correct configuration bits to the ADS8344, wait for the conversion to be done and then read the data from it. The channel selection is done in this control register, in bits 6, 5 and 4 as follows.

Table 5.3 Module #2 ADC inputs review and selection.

Channel number	Circuit	Bit 6	Bit 5	Bit 4
CH0	Pressure #1	0	0	0
CH1	Pressure #2	1	0	0
CH2	Left CHT	0	0	1
CH3	Right CHT	1	0	1
CH4	Left EGT	0	1	0
CH5	Right EGT	1	1	0
CH6	N/C	0	1	1
CH7	N/C	1	1	1

The control register of the ADS8344 is very simple to use. After receiving the 4th control bit, it starts acquiring the actual channel input signal, and after receiving the LSB, it does Successive-Approximation bit decisions and immediately puts them at $D_{OUT.}$

5.8 I/O RS-232 interface

Two interfaces are available in EFMS, RS-232 serial port and Ethernet. At the presentation date, Ethernet is not functional yet, but will be further implemented. For a correct EFMS operation checking, RS-232 port can be used for the moment.

The dsPIC33FJ256GP710 brings a pair of built-in UART modules, which can be used to transmit and receive data using the RS-232 protocol. It just needs a RS-232 transceiver such as the MAXIM MAX3232 between the UART ports and the D-SUB 9 connector.

The MAX3232 is a 2-channel RS-232 transceiver, which can be supplied from 3.0V to 5.5V and can run up to data rates of 120 kbaud. It just needs four small 0.1µF external capacitors. This RS-232 transceiver is using the primary UART module of the dsPIC33. Hardware handshaking lines (RTS and CTS) are implemented, but for the moment, are not used by the program application.

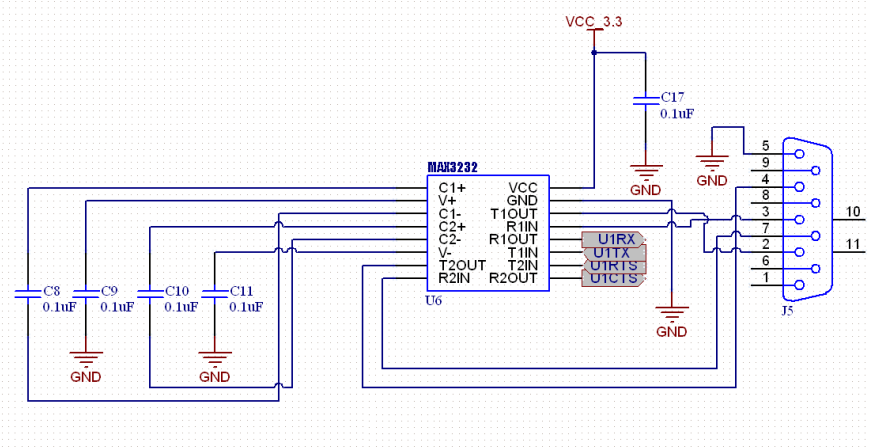


Fig. 5.25 EFMS RS-232 interface circuit.

5.9 Power source conditioning

EFMS will be fed from the batteries unregulated +12V available all around the plane. A power conditioning stage is required to obtain regulated +5V, +3.3V and +2.5V for all the circuits and silicon ICs within the EFMS.

This is done by cascading three LDO (Low-Drop-Out) regulators. LDO main characteristics are ease-of-use, low noise and low cost.

The EFMS uses these National Semiconductor[6] LDOs to obtain the required voltages:

12 to 5.0V	5.0V to 3.3V	3.3V to 2.5V or 5.0 to 2.5V
National Semiconductor LM2940-5.0	National Semiconductor LP8340-3.3	National Semiconductor LP3871-2.5

LDO require external capacitors to assure stability. Usually two capacitors are used per each LDO: one at the input and one at the output.

LM2940 and LP3871 support a maximum current of 1A, while LP8340 supports 0.8A. Nevertheless, maximum current consumption for EFMS is estimated to be about 400mA as much. MCU consumes 80mA working at maximum power (and for the moment, it will not), whilst other devices such as amplifiers, transceivers and sensors are designed to consume low currents. Even so, this permits using the LDO without any heatsink installation.

Fig. 5.26 shows this power conditioning stages.

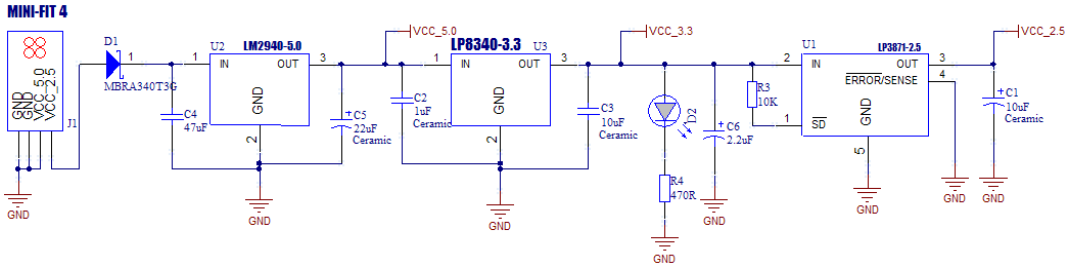


Fig. 5.26 Module #1 power conditioning stages. Note that Module #2 has the same layout, but no 3.3 LDO is present. 5.0 LDO output is directly connected to 2.5 LDO input.

5.10 EMI protection

EMI stands for ElectroMagnetic Interference, and is an external unwanted disturbance that will affect any circuit, up to the point of disrupting the on-circuit signals. Typically, these interferences are created by external objects or effects, natural or artificial, carrying rapidly-changing electrical currents. For instance, in an UAV frame, the explosions within the engine cylinders cause a huge amount of electrical noise, creating a really harshly environment.

The EFMS has been designed in a way that its circuitry removes almost all possible noise to it keep away from the sensor signals. In the acquisition stages, the devices which are directly in contact with the sensor terminals are low-noise and especially designed to remove spurs, and in some cases, external filtering and protection circuits are used. The enclosures are also kind of EMI protected.

5.10.1 Conductive interference reduction

Conductive interferences are mostly caused because the interference source and the 'victim' system share some kind of connection. In most cases this can be found on source, ground and mass connections.

The most typical interference is caused by ground loops. These appear by the voltage difference between two ground connection points that are away from each other and have same system circuits connected to each. In EFMS, possible ground loops are avoided, by having a unique ground connection at Module #1, which is common with the ground plane at Module #2. Module #2 is not connected to another ground (Fig. 5.27 left).

5.10.2 Capacitive and inductive interference reduction

The best form for cancelling capacitive interferences is using shielded wires, connections and enclosures. However, connectors are hard to protect.

External wiring to sensors or between Modules is done using high-quality top-end shielded coaxial wires. Because Module #1 has the main ground

connection, all existing external wire shields are only connected to it, as shown in Fig. 5.27 right. This avoids current loops on the shield and on the wires, which would probably lead to interference currents on it and signal distortion.

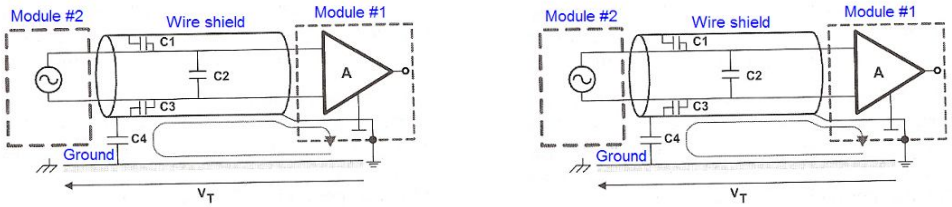


Fig. 5.27 Avoiding ground loops (left) and wire shield (right) loops.

SECTION 6. SOFTWARE DEVELOPMENT

6.1 Software development stages

The EFMS software development has been done in two separate parts: microcontroller program stage and MAREA implementation stage. By considering that EFMS would not be limited to be used only with the UAV Shadow, the microcontroller programming was started first and about one month later MAREA engine-related services revision and enhancement was then commenced as well.

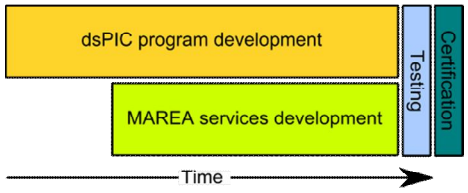


Fig. 6.1 Programming distribution in time.

At the time of writing this documentation, MAREA services are functional, but not of the required robustness (see “6.2 Functionalities and requisites” below, and “SECTION 7. FINAL BALANCE”).

6.2 Functionalities and requisites

Software is divided in two ‘programs’: device side (it will be referred as EMU, Engine Management Unit) and MAREA side. Its functionalities and requisites are shown below.

6.2.1 EMU side

Open-source, easy to understand program code

Program code in the microcontroller must be programmed in a smart, clear way. Its code arrangement must be intuitive and easy to read, and all functions, methods, variables, etc… should be adequately documented.

Less than half-second iterative data acquisition

The EMU must obtain the data from all the inputs, process it and store it inside the device memory at a rate equal or less than 500ms, which has been

determined to be the minimum refresh rate required for RPM display. The other data readings are less important, as they will not fluctuate as fast as RPM.

Real-Time Clock for a future Black-Box system

EFMS will further implement a Black-Box system, by implementing an external SPI EEPROM memory connected to one of the expansion ports. It will store all data from sensors, with a time stamp. This forces the use of a time source, whose actual time will be refreshed at each system start-up. It is services side duty to refresh this time.

Easy access to sensor data

The EMU must provide an easy and simple access to the most current data. The device refreshes the data registers at a rate of less than 300ms, and it must be always accessible via RS-232 or Ethernet, in a *request-receive* form ().

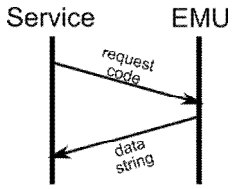


Fig. 6.2 *Request-Receive*.

Program execution robustness

The program must avoid incorrect set-ups, buffer overflows, memory leakages, or endless loops that could eventually lead to address, stack and or math errors that would freeze the device.

Robustness after device crash

In case of a sudden crash or freeze, the device must be capable of always re-initializing, re-configuring itself and restarting the program without human intervention.

6.2.2 Services side

There are two separate services that manage engine information. They work together, with constant dependencies, but as separate threads.

In short, it can be said that one of them is responsible for taking the data from EMU and storing it inside known data containers. It can be understood as an airborne system, as it is in direct contact with an UAV system. This service is known as *EngineManager*.

The other service is known as *EngineMonitor*. It repeatedly takes the data from these data containers, processes it and shows it into Flight Monitor. Its most important feature is the early-alarm system. This service is more like in the ground station, as it shows the data to the UAV controller.

Create and maintain connection with the device

The *EngineManager* has to create and maintain a correct RS-232 or Ethernet connection with the device. Eventually it has to check if the EMU is still properly connected. In case of data corruption or timeout, it has to check if the EMU is still connected or working.

Request-Receive data acquisition

Interaction between the device and the *EngineManager* service is accomplished in a *request-receive* form. This procedure is explained in "6.5 Data packets format".

Half-second rate iterative data acquisition

When properly connection is accomplished, the *EngineManager* service must keep listening to the device data, at a fixed rate of 500 milliseconds. It should then acquire it, process it and pass it on to storage registers.

Easy to edit configuration file

An XML configuration file must exist within *EngineMonitor* service. This configuration file must contain all ranges, units and other useful information for each sensor or monitored parameter. It must be easy to locate this XML file, in order to correctly edit it.

Early-alarm system

The *EngineMonitor* service must obtain the data from the *EngineManager* storage registers at a rate of 250ms. Within this time, it must process this data. That means to compare it with the information contained inside the XML configuration file and then refresh the Flight Monitor and the Early-Alarm system.

6.3 EMU program architecture

The EMU program is single-threaded. When a device Power-On is done, the program initializes it and its peripherals and then enters an endless loop, which iteratively checks the inputs and processes data. It takes less than 250ms to accomplish a complete refresh.

Flowcharts to this program and explanations to each process can be found in the next subparts.

6.3.1 Main program

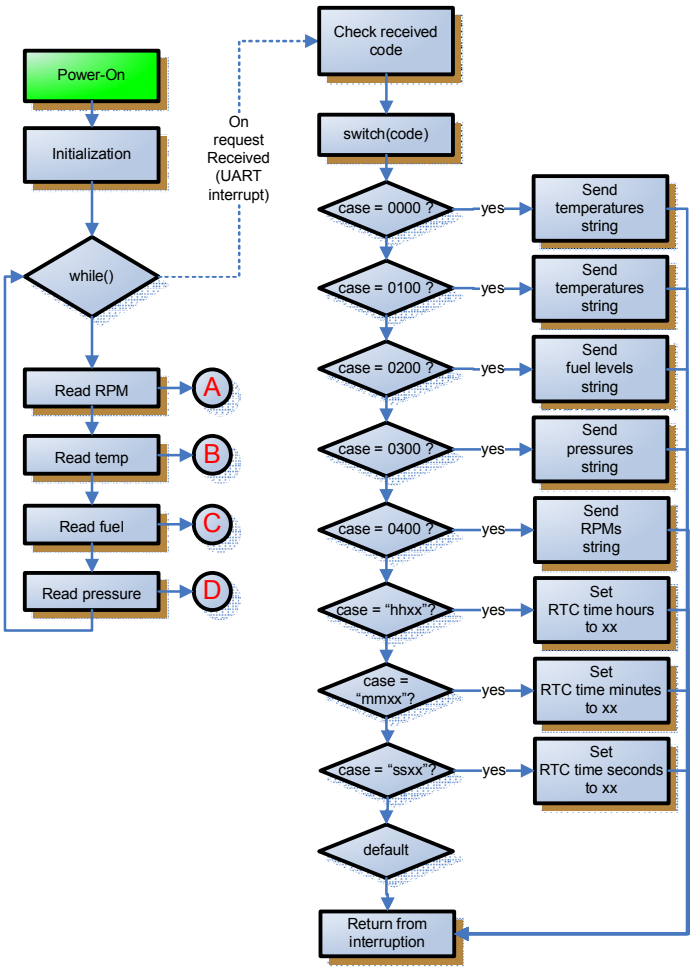


Fig. 6.3 Main program flowchart.

This is a simplified main program flowchart. As required in "6.2 Functionalities and requisites" part, it keeps reading the inputs, but does not actually export the data. It just stores the latest acquired data into internal registers. It only exports data when requested via RS-232 (if Ethernet was used, as it will, it would work in the same manner). This is done to prioritize data acquisition and storage instead of its continuous export.

The request codes can be also seen in the flowchart, and are standardized. This means that they are always the same. The output data strings are also in a predefined format. All this enforces the services side to use this required codification and to adapt to the output strings format. More details to these codes and formats are explained in "6.5 Data packets format" part.

6.3.2 RPM acquisition

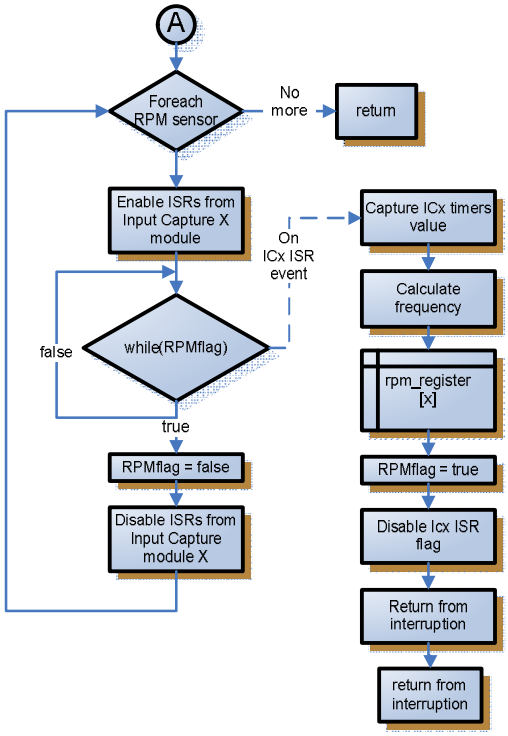


Fig. 6.4 RPM calculation routine.

As told in “5.5 Sensors: RPM from engine and wheels” part, RPM acquisition is done using the Input Capture (IC) modules from the microcontroller. These cause an interruption when two consecutive rising edges belonging to pulses from the Hall-Effect sensors in the engine and wheels are received. Because the engine is considered to turn at a maximum of 10000 RPM, this means that an interruption will be caused every 100µs as much. This would cause the program to keep in these interruption routines much more time than it is necessary, and would slow down the other routines execution.

To avoid this problem, IC modules interruptions are only enabled temporarily and sequentially. When a single reading and calculation of RPM is done and stored into the register, the corresponding IC module interrupt is disabled and the next is enabled. This is done up to four times, for each IC module. After that, a return instruction to the main program is done.

6.3.3 Temperature and pressure acquisition

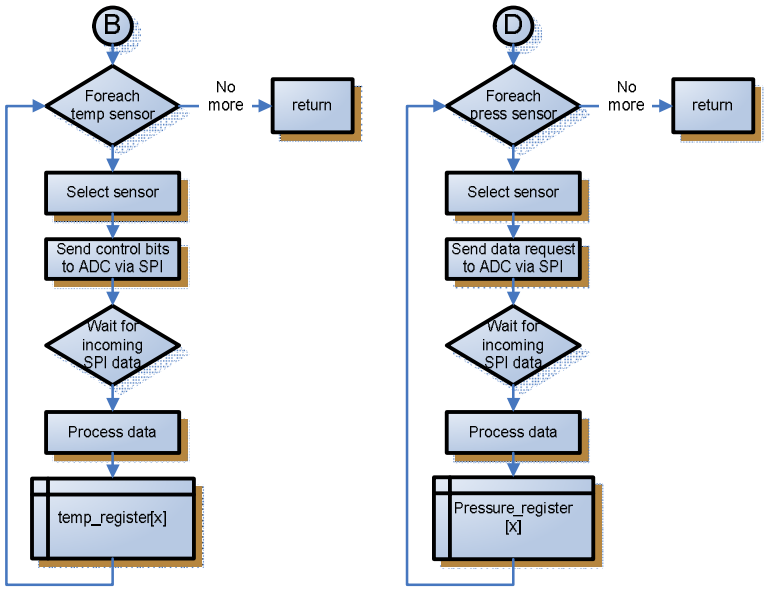


Fig. 6.5 Temperature (left) and pressure (right) acquisition routine.

Temperature and pressure conversion to digital words is done externally by a 16-bit ADC (ADS8344), which communicates with the microcontroller via SPI port. A single 8-channel ADS8344 is used in the whole system. Channels are selected by control bits 6, 5 and 4, so for each temperature, a different control byte has to be sent to the ADS8344.

When control bits are sent and after about one more SPI clock, the ADS8344 sends the selected channel conversion result.

In case of temperatures, some processing is necessary. For example, thermocouples data need further calibration, which is accomplished both with isothermal temperature reading and calibration tables (see “5.4.1 Thermocouples: CHT and EGT” subpart).

After processing, resultant data is stored in corresponding registers.

6.4 Services architecture

As told in some parts of this documentation, MAREA middleware uses a highly flexible services-based architecture (SOA). The avionics systems are composed of a set of distributed elements, known as *services*, which operate on “top” of MAREA. *Services* are the unit of work, implemented and offered by a *service provider*, in order to achieve final results for a *service consumer*.

Because of its enormous flexibility, SOA permits reusability of services. This means that at the time of creating a new system, one can take an existing generic service and mold it.

All of this creates a network of cooperating systems. MAREA uses a kind of *publish/subscribe* model for sending and receiving data, events and commands among these services. Some services produce data, so they are publishers, whilst others, subscribers, just take and use this data. MAREA is responsible of starting, stopping and monitoring these services. When there is a change in any service behavior, it notifies the dependant systems.

All services are addressed by a name, and the middleware discovers the real location of the named service. This feature allows the programmer not to know where the service resides. He just has to care about programming its functionalities and giving it the right name.

As told, EFMS uses two services. Their implementation within MAREA architecture is shown below in

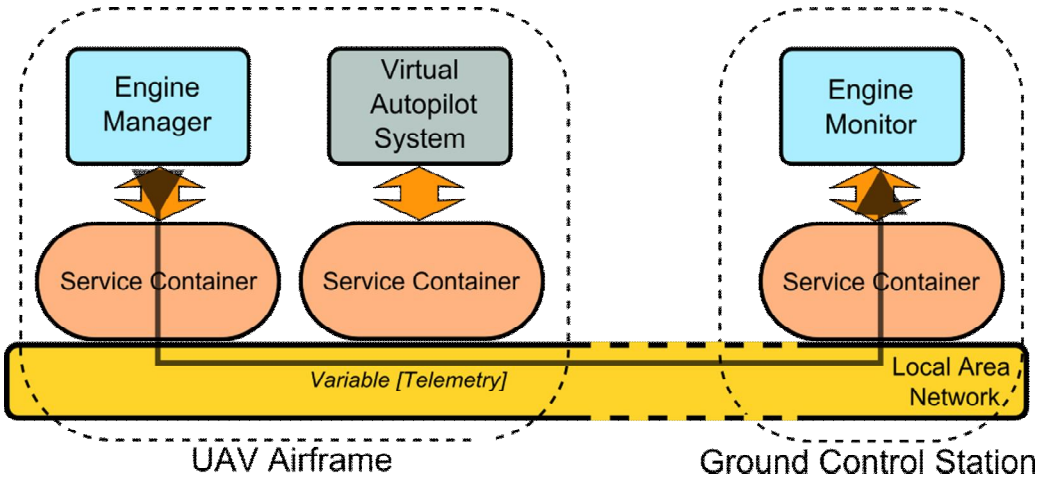


Fig. 6.6 Middleware Architecture.

As shown in Fig. 6.6, the services are executed and managed by service containers, which are unique within the network. Every service container can manage several services and provide common functionalities (network access, local message delivering, name resolution, etc…) to its internal services.

6.4.1 Service layers

Many of the services are double layered. This means that they have a layer interacting with MAREA, and another layer interacting with the data source element (the EMU in this case). These layers should be independent from each other, so changing the EMU system for other, would be transparent on the MAREA side layer.

This allows creating general service architecture with no real dependence with the hardware it is connected to. If this hardware is replaced by a different one, the system should not be noticed. It is the upper layer which has to deal with the specific EMU system, so only its program code might be changed if the EMU was replaced. This proposal is shown in Fig. 6.7.

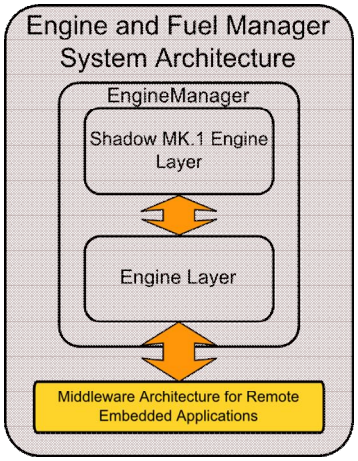


Fig. 6.7 EngineManager layer utilization example.

The interaction between these independent layers is accomplished by the use of interfaces. The upper layer may adapt to this interface requirements to export its data to the lower layer.

An example of usefulness of layers proposal is shown in Fig. 6.8, where different Engine Manager Units can be used over the same *EngineManager* lower layer.

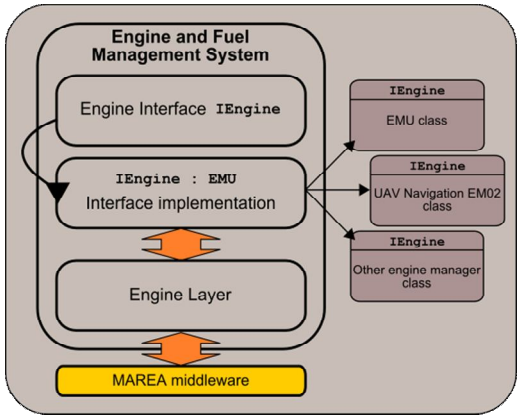


Fig. 6.8 EFMS Interfaces composition.

6.5 Data packets format

As told before, accessing to the EMU via RS-232 is easily done by sending a code to it and awaiting its response. When the EMU detects a known code, it takes the newest available data from the corresponding register, puts it in a specified format and sends it.

6.5.1 RS-232 configuration

The RS-232 port in the services side must be set-up matching the EMU configuration. This configuration is as follows, and cannot be changed:

Baud Rate	Handshaking	Parity	Data bits	Stopbits	End of String designator
56000	None	None	8	1	'\0' NULL

RS-232 protocol is prone to cable length. The connection between the EMU and the computer handling MAREA must be kept as short as possible.

6.5.2 Available request codes

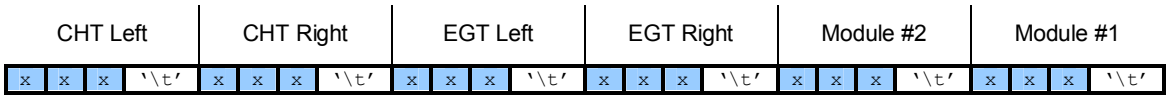
Codes are sent as ASCII codified strings. EMU data is stored in registers according to its data type. Each request will return a string with this data.

Code	Primitive type	EMU response
0000	ASCII string	"ack"
0100		Temperatures
0200		Fuel levels
0300		Pressure levels
0400		RPM

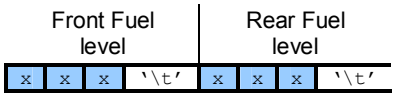
6.5.3 Received data packets format

Received strings contain each sensor result separated between them by a Tab character '\t'. These strings are composed of ASCII characters.

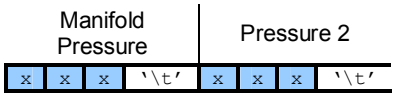
Temperatures string format



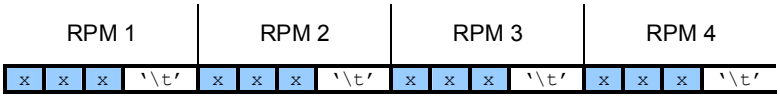
Fuel level string format



Pressure levels string format



RPM string format



Data extraction is then easily done using C# *split* function.

Note that not all the software programs would use all these parameters. In fact, *EngineMonitor* service and the Flight Monitor screen only stare for one CHT and one EGT, whilst EMU is sending one CHT/EGT parameter for each cylinder.

Here, service layers (see 6.4.1 Service layers) play the role: the upper layer (which is “in touch” with EMU) gets the Temperatures string, parses it and recognizes the two first resulting strings as `CHTL` and `CHTR`. Then it takes and means them, and sends the result to the interface, which requires just one EGT/CHT temperature for the lower layer (in contact with MAREA) needs.

Hence, if the EMU was changed by another system, only the upper layer MAREA *EngineManager* was to be modified and adapted to it, whilst the other services would remain unchanged.

SECTION 7. FINAL BALANCE

7.1 Conclusions

The main objective of this project was to design an Engine and Fuel Monitoring System but not to build it. Its production and the MAREA middleware implementation were considered after a period of time, because of the high interest I showed on this project. As a result, the efforts were soon focused to this because of the importance it had it application onboard the Icarus UAS Platform.

MAREA implementation was supposed to have been a relatively easy and quick workout. However, at the presentation date, Middleware implementation has not been completed yet. This is basically because MAREA is still an under-development platform, and the engine-related services, which were supposed to be more or less functional, were not. The EMU adaptation to MAREA has brought some problems lately, which have been delaying the release of a stable version. But on the other hand, this problem will certainly be solved in the next weeks.

EMU electronics soldering and testing have been delayed to some weeks after the presentation is done. This decision has mainly been taken because having the circuits ready within the presentation date was not a priority. EMU circuits were supposed to be ready for the due date, but PCB design took too long, and also some problems with the soldering responsible person have delayed the circuit availability too much.

However, the general balance is excellent. An entire engine monitor system has been designed (this was the main objective), and almost built. Only the last few steps of a functional project are left to do: soldering, electrical testing, reprogramming and certification. The main aim was to do a project, not a final product. But finally, a quasi final product has been obtained.

7.2 Costs

Information on costs can be found on Section 4 of the “Anexos” document.

7.3 Future lines of work

Future expansions for the EFMS could be:

- Complete and functional MAREA implementation.
- MAREA implementation via Ethernet connector.

- Enhancement of the sensor data gathering, give more precision to the readings.
- Black-Box function.
- Stand-alone program for data extraction for other non-MAREA applications.

My aim is to continue working together with the ICARUS team, as I am going to start Technical Aeronautics Engineering.

Hence, some of these future proposals will be implemented by me in the next weeks after the presentation date, as they are included and necessary for the ICARUS UAS Platform development and flight.

7.4 Environmental care

By the time the ICARUS UAS Platform gets ready to fly, its main application will be the detection and control of forest fires. Catalunya countryside is a hot-spot for these fires, as it has a warm, hot climate, especially in those summer months.

ICARUS UAS Platform comes to replace the manned helicopters and airplanes that are currently used for fire-awareness purposes. These are high fuel consumers when compared to the fuel consumed by a small UAV engine, and its propellers are far noisier.

These are specific EFMS environmental care issues:

- Engine monitoring brings to the controller better control on the engine efficiency. This allows to maintain a correct combustion on it, and thus to save fuel.

- With the alarm system, UAV crashes can be avoided. Accidents and the removal of the wrecked UAV from the crash zone could cause such a big impact on the environment.

- EGT also affects the noise levels of the engine and on the exhaust CO^2 levels. Its reading directly indicates if an incorrect combustion is happening [7].

- All EFMS electronic components are RoHs compliant, which means that meet the restrictions on use of six hazardous materials in its manufacture, which are Lead, Mercury, Cadmium, Hexavalent chromium (Cr^{6+}), Polybrominated biphenyls (PBB) and Polybrominated diphenyl ether (PBDE). More information on RoHs compliances can be found on [10].

SECTION 8. BIBLIOGRAPHY

8.1.1 Books, articles and application notes

Pérez García, M.A.; Álvarez Antón, J.C.; Campo Rodríguez, J.C.; Ferrero Martín, F.J.; Grillo Ortega, G.J, *Instrumentación Electrónica*, Thompson, Madrid 2005.

Pastor E., *"Engine and Fuel Monitor Service Design, Functional Specifications"*, 5 pages, 2007.

Royo, P.; López, J.; Santamaria, E.; Barrado, C.; Pastor, E., *"Virtual Autopilot System for the integration of UAS civil applications"*, *Journal of IEEE Transactions on Aerospace and Electronic Systems*, Vol. 6, Nº 1, 21 pages, January 2007.

UAV Navigation, *EM02 User Manual*, Revision 03

Baker, B., *Temperature Sensing Technologies*, AN679, Microchip Technology Inc.

Baker, B., *"Precision Temperature Sensing with RTD Circuits"*, AN687, Microchip Technology Inc.

Baker, B., *"Single Supply Temperature Sensing with Thermocouples"*, AN684, Microchip Technology Inc.

Baker, B. *"Anti-Aliasing, Analog Filters for Data Acquisition Systems"*, AN699, Microchip Technology Inc.

Julicher. J, "*Simplified Thermocouple Interfaces and PICmicro MCUs"*, AN844, Microchip Technology Inc.

Lepowski, J. *"Temperature Measurement Circuits for Embedded Applications"*, AN929, Microchip Technology Inc.

Palmer, M. *"Using the CCP Module(s)"*, AN594, Microchip Technology Inc.

Klopfenstein, Rex, *"Software Linearization of a Thermocouple"*, *Sensors*, December 1997.

Simpson. C., *"Linear and Switching Voltage Regulator Fundamentals"*, National Semiconductor.

Microchip Technology Inc., *"dsPIC33FJ256 Reference Manuals"*, Microchip Technology Inc.

8.1.2 WebPages

[1] Integrated Dynamics: http://www.idaerospace.com/

[2] UAV Navigation: http://www.uavnavigation.com/

[3] Microchip Technology Inc.: http://www.microchip.com/

[4] Texas Instruments: http://www.ti.com/

[5] Freescale Semiconductor: http://www.freescale.com/

[6] National Semiconductor: http://www.national.com/

[7] Information on CHT and EGT application: http://www.foxvalleykart.com/egt.html

[8] ISSPRO fuel level sensors: http://www.isspro.com/products.php?cat=99

[9] Westach EGT: http://www.aircraftspruce.com/catalog/inpages/westegt7.php

[10] RoHs: http://www.rohs.gov.uk/

SensorsMag: http://www.sensorsmag.com/

Embedded systems: http://www.embedded.com/

Analog Devices: http://www.analog.com/

Wikipedia: http://www.wikipedia.com/

Escola Politècnica Superior de Castelldefels
UNIVERSITAT POLITÈCNICA DE CATALUNYA

ANEXOS

TÍTULO:	**Engine and Fuel Manager System for Unmanned Aerial Vehicles**
AUTOR:	**Julio Sagardoy Pérez**
TITULACION:	**Ingeniería Técnica de Telecomunicaciones, especialidad en Sistemas de Telecomunicación**
DIRECTOR:	**Enric Pastor Llorens**
FECHA:	**21 de Mayo de 2009**

CONTENTS

SECTION 1. INTERNAL COMBUSTION ENGINES

In this section, the basis of generic 2-stroke engines will be studied. It is interesting to whip trough it because UAV Shadow uses a 2-stroke 250cc boxer engine, which will be analyzed as well.

1.1 The 2-stroke engine

Here is a schematic of a basic 2-stroke engine with its parts:

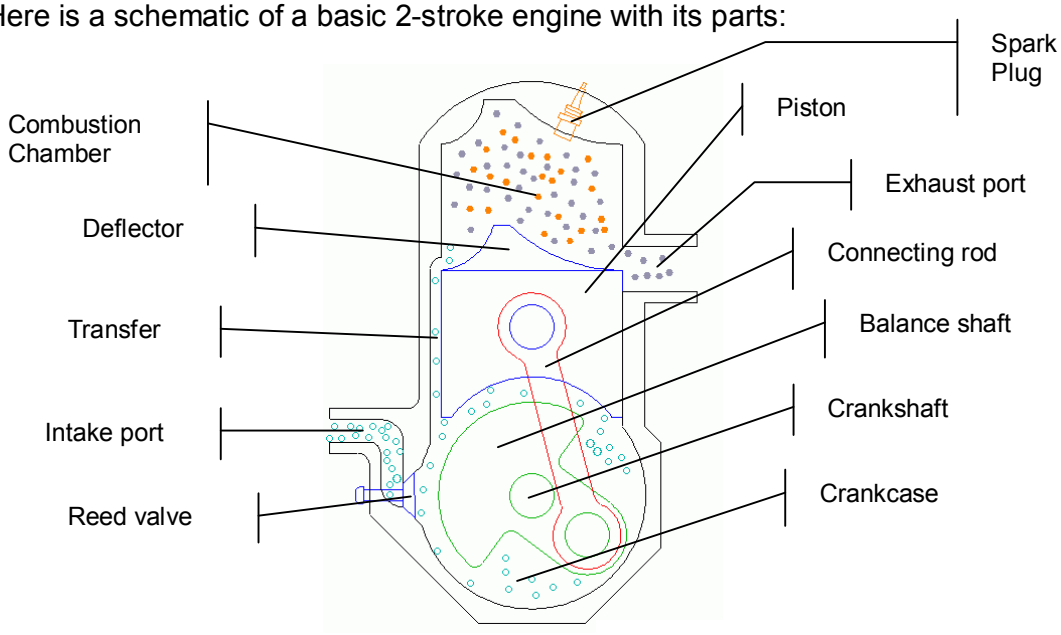


Fig. 1.1. Generic section of a 2-stroke engine.

The main characteristic of 2-stroke engines is that they complete an entire cycle in just one spin of the crankshaft. A cycle has four different stages which are the intake, the compression, the combustion and the exhaust stage. The basis of the 2-stroke engine is to do the intake at the beginning of the compression stroke, and the exhaust function at the end of the combustion stroke.

This is the process it follows:

- Intake: An uncompressed mix of fuel, air and oil which comes from the carburetor is sucked into the crankcase. This occurs because the piston is going upwards at this moment and the intake port is uncovered (or valve-opened, because some designs have some kind of reed valve here), so the vacuum which is being created into the crankcase suctions it.

- Compression: Almost at the same time that intake is happening, the piston compresses the preceding mix that was already in the combustion chamber.

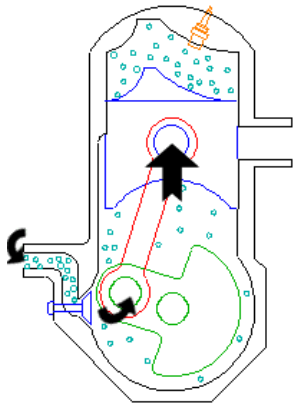


Fig. 1.2. Intake and compression stages.

- Combustion: When the piston is at its highest point, the compressed mix is ignited by a spark created by the glow plug. This causes a violent explosion of the mix, which propels the piston downwards.
- Exhaust: The burnt mix is expelled from the combustion chamber by the time the piston lets the exhaust port uncovered. Short after, the transfer port is uncovered too, so the first mix in the crankcase goes through it and gets into the combustion chamber.

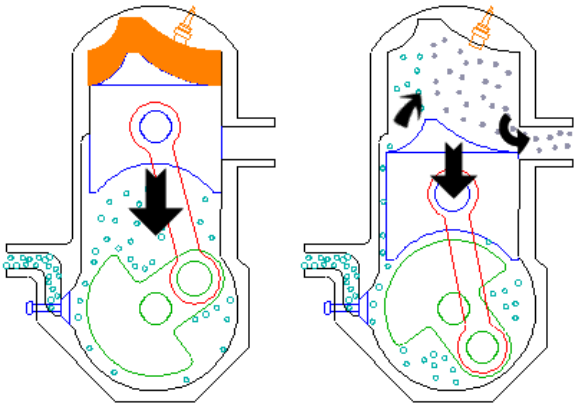


Fig. 1.3. Combustion and exhaust stages.

The process is then repeated again.

The engine shown uses a ‘scavenger’ piston. This is a technique that uses a deflector to direct the intake mix to the upper part of the combustion chamber thus forcing the exhaust gases down to the exhaust port. This is used to avoid fresh mix to slip through the exhaust port hence losing efficiency. Some better techniques are available; this one is known to be less effective, because the

deflector makes the piston heavier and also reduces the effectiveness of the combustion chamber.

1.2 Engine layouts

Engine behavior varies depending on both the number of cylinders and its arrangement.

1.2.1 Number of cylinders

The engine scheme shown in the figures before is a single cylinder set-up, and it is the simplest layout possible. They feature a small, simple and economical construction. However, between the time after and before a combustion stage, the engine is running because of the inertial force on the flywheel, and therefore is not producing new energy.

This no-productivity gap is compensated by using more cylinders, so drawing an offset between each piston position and therefore alternating the explosions can lead to a smoother running of the engine. Consequently, a more efficient use of the cycle time will be happening, so it will be producing more power per time unit. More cylinders will also give the engine better power to weight ratio.

For example, in a 2-stroke, two cylinder engine, this offset could be 180º, so when one of the pistons is in its lowest position (beginning of the intake/compression stage) the other will just be in the combustion stage, and when this happens its descending force will help the other to compress. This configuration may use two separate balance shafts. Several other offsets are used. In Harley-Davidson motorbikes, V-twin engines are used, and both pistons are connected to the same pin in the crankshaft, so the offset is minimum, and the explosions are almost one after the other (it is easy to note this, just by hearing how it sounds). This creates a less balanced engine, with more vibrations, but provides more torque.

1.2.2 Position between cylinders

Position between cylinders affects directly to the engine global shape and behavior, however it does not affect the global performance that much. Most common configurations and its characteristics are these:

- Straight (in-line): Cylinders are mounted straight in a line, side by side and the pistons drive the same crankshaft. It is the most extensively used layout.

Fig. 1.4. Straight / in-line layout.

- V: Cylinders are aligned too, but a half of them are in a separate plane, forming an angle respect the others, so they appear to be forming an V when looking them along the axis of the crankshaft. The pistons drive the same crankshaft. They can be connected to the same pin in crankshaft (see picture below) or to separate pins.

Fig. 1.5. V layout.

- Flat (Boxer): All cylinders are in horizontal position, but half of them are in opposed position and the pistons are all driving the same crankshaft. It is called boxer because the piston movements seem to fight each other.

Fig. 1.6. Flat / boxer layout.

In-line shape is the most extended. It is very easy to be produced, because the engine block is done from a single piece. Due to the cylinders standing one next the other, these engines tend to be larger compared to the other two arrangements. They are also known for being a little ‘rude’ in behavior.

Boxer shapes are known to have better characteristics, because they are used to be positioned in horizontal position, so the overall height of the engine is less than an inline set-up. Because of this, lower gravity center is achieved. And due to opposed and alternating piston explosions, vibrations are reduced, so these engines are known to be very smooth. However, they are harder to build. The engine block is built and assembled in two pieces, and usually two separate feeding systems need to be used, one for each group of cylinders. This leads to more complexity, and by consequence, more expensive and more prone to mechanical failure.

V-layouts are a compromise between in-line and boxer, and are the most used in larger engines. Depending on cylinders number, a V-layout can result in a more or less balanced engine, compared to an in-line with the same number of cylinders. This happens whenever the number of cylinders in each bank is odd.

SECTION 2. UAV Shadow powerplant

The current UAV Shadow engine is a cast-aluminium 2-stroke, 2-cylinder boxer engine, air-cooled, fed by a butterfly-controlled carburetor and has a total capacity of 239 cubical centimeters. It runs well with 95-octane unleaded fuel, although 98-octane is preferred. Oil mix is necessary, and must be done manually when refueling, with a ratio of 1:50 to 1:80.

Its air to fuel mix can be adjusted by two knobs, each one varying this mix at high RPMs (butterfly full-opened) and at low RPMs (butterfly closed, minimum air flow). These two knobs must be adjusted when at ground, and must ensure a proper engine running in all RPM range avoiding both overheating and engine stall due to fuel flooding.

It features an electronic ignition module, a generator-alternator system, two magnetos per cylinder and a built-in Hall-Effect sensor for the electronic ignition.

It is made by *3W Modellmotoren*, a German manufacturer of remote control planes and engines. The model specification is 240iB2.

Fig. 2.1. 3W 240iB2 engine used in UAV Shadow.

Table 2.1. Engine technical specifications.

Cylinder capacity	239cc
Power	22CV
Bore diameter	57.5mm
Stroke	46mm
RPM range	1000 – 7500 RPM
Spark plug	NGK CM-6 (∅=10mm)
Weight	6.9kg
Fuel	Unleaded fuel, 98 Octane is optimal
Oil	Fully synthetic
Oil mix ratio	1:50 – 1:80

SECTION 3. Resistance Temperature Detectors

Metals have the capability of increasing its electrical resistance by the time its internal energy raises. This property has been widely used from long time ago in the design and the build of RTD, whose main element is a thin metal film.

Its electronic symbol and its external appearance are shown in Fig. 3.1:

Fig. 3.1. Electronic symbol and external appearance of an RTD.

The next table shows common RTD metals resistivity and thermal coefficient of the resistance variation as a result of ambient temperature changes.

Table 3.1. Common RTD metals.

Metal	Resistivity (ρ), [Ω, m]	Thermal coefficient (α), [K^{-1}]
Platinum, Pt	$10.6 \cdot 10^{-8}$	$3.9 \cdot 10^{-3}$
Nickel, Ni	$6.84 \cdot 10^{-8}$	$7 \cdot 10^{-3}$
Wolfram, W	$5.6 \cdot 10^{-8}$	$4.5 \cdot 10^{-3}$
Copper, Cu	$1.68 \cdot 10^{-8}$	$4.3 \cdot 10^{-3}$

Platinum has the lowest thermal coefficient while nickel has got the highest. In practical terms, lower thermal coefficient will have a lower sensibility and vice versa. So nickel will increase its resistance as temperature rises faster than platinum would. However, having platinum more resistivity, thin wires with a notorious resistance can be obtained without being lengthy. This is important because metals get larger with temperature, and resistance is given by:

$$R = \frac{\rho \cdot l}{A}$$

Because of this, in RTDs where nickel is used, changes in its dimensions are more notorious, and this leads to non-linear changes in resistance-temperature characteristic. That is why not all metals are used in RTD.

Applications

Thermocouples cold junction compensation, bridge temperature, calibration issues and process control.

Static transfer function

Given by the manufacturer, shows the relationship between input parameter (temperature) and its output (resistance).

Fig. 3.2 shows a portion of the static transfer function of platinum RTD known as PT-100. The resistance value for 0ºC is known as R_0, and in this case it is 100Ω, hence the PT-100 designation.

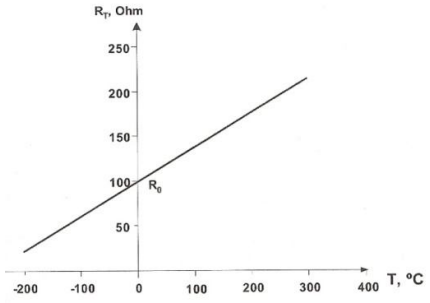


Fig. 3.2. Static transfer function of a platinum RTD.

The nominal slope value for RTDs is given by the IEC (International Electrotechnical Commission) and it is fixed to 0.385Ω/ºC for PT-100. For a PT-1000, R_0 will obviously be 1000Ω. Hence its sensibility is fixed to 385Ω/ºC, which makes them have more immunity to resistance changes in wiring.

The next equation shows the relationship between temperature and resistance:

$$R_t = R_0 \cdot [1 + A \cdot t + B \cdot t^2]$$

Where:

A = 3.9083E-3ºC^{-1}

B = -5.775E-7ºC^{-2}

Tolerance classes

RTD sensors are divided into classes according to their limit deviations (see **Table 3.2**).

Table 3.2. Tolerance classes

Class	±limit deviations in ºC
DIN A	0.15 + 0.002 · |t|
DIN B	0.3 + 0.005 · |t|
2x class B	0.6 + 0.005 · |t|
1/3 class B +	0.1 + 0.0017 · |t|
1/3 class B -	0.1 + 0.005 · |t|

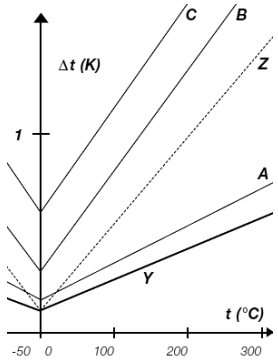


Fig. 3.3. Tolerance classes.

Conditioning circuit for RTD

Measuring temperature implies measuring RTD resistance, then evaluating it through its static transfer function. This does not provide an easy-to-measure electric signal proportional to the temperature. Instead of this, feeding RTD with a known current (limited by the self-heating) will result in a voltage through it which will be proportional to the temperature.

Floating current source

For best linearity, a floating stable current source might be designed, so it feeds the RTD with a constant current which will not vary whatever the load is. It is the voltage drop across the RTD which is going to vary.

This method, however implying more circuit complexity, gives the best results in all aspects, and it is the one to be used in the design of the UAV Shadow's temperature acquisition.

Fig. 3.4 shows a possible implementation, where a voltage reference and two operational working as 0-gain differential amplifiers are used to generate a 1mA current source.

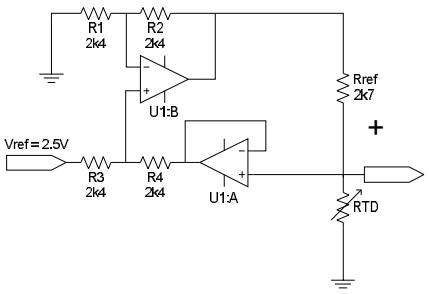


Fig. 3.4. Floating current source feeding RTD in UAV Shadow.

Here:

$$V_{OUT\_U1:B} = V_{REF} \cdot G_{U1:B} + V_{OUT\_U1A}$$

- $V_{OUT\_U1:B}$ is the U1:B amplifier output voltage.
- $G_{U1:B}$ is the gain of the U1:B amplifier, which equals 1.
- $V_{OUT\_U1:A}$ is the U1:A amplifier output voltage.

Because the unity gain in U1:A and assuming that $V_{OUT\_U1:A} = V_A$:

$$V_{RREF} = V_{OUTU1:B} - V_A$$

$$V_{RREF} = V_{REF} = 5V$$

Current $I_{RREF} = V_{REF} / R_{REF}$, is constant and independent from the voltage in A, which will correspond to the voltage drop across the RTD. I_{RREF} is limited by the RTD self-heating desired value. V_{REF} is taken from the 5V regulated voltage source.

Labfacility PT-1000 RTDs feature less than 0.5ºC/mW of self-heating. If reference current, *I*, is fixed to 1mA, the maximum temperature error due to self-heating will be:

$$P = I^2 \cdot R = 0.001A \cdot 2800\Omega = 2.8mW$$

$$\Delta T = 0.5℃ \cdot 2.8mW = 1.4℃$$

which is insignificant.

Note that this circuit would also work to other resistance measurement applications.

SECTION 4. COSTS

Costs list has been restricted to the components that have been used in the EFMS development. It does not include PC, software, software licenses, and all items that form a part of the built workplace and that are for free laboratory use.

4.1 EMU Bill-Of-Materials

Element	Used Qty.	Unit Price (mean)	Price	Acquired as sample?
ADS8344 Analog to Digital Converter	1	EUR 17.00	EUR 17.00	S
Amplifiers, instrumentation amplifiers	7	EUR 7.00	EUR 49.00	S
Connectors, headers, etc	66	EUR 0.40	EUR 26.40	EUR 26.40
dsPIC33FJ256GP710	1	EUR 7.00	EUR 7.00	EUR 7.00
ENC28J60 Ethernet transceiver	1	EUR 2.50	EUR 2.50	EUR 2.50
BOSS Enclosures	2	EUR 15.00	EUR 30.00	EUR 30.00
MAX3232 RS-232 transceiver	1	EUR 2.00	EUR 2.00	S
SMD Capacitors (various values)	39	EUR 0.21	EUR 8.19	EUR 8.19
SMD Schottky diodes	2	EUR 0.50	EUR 1.00	EUR 1.00
SMD Resistors (various values)	41	EUR 0.04	EUR 1.64	EUR 1.64
Special connectors (RS-232, Ethernet, ICD2…)	5	EUR 6.00	EUR 30.00	EUR 30.00
TMP121 temperature sensors	2	EUR 1.50	EUR 3.00	S
Voltage regulators	5	EUR 1.50	EUR 7.50	S
XTAL	3	EUR 4.00	EUR 12.00	EUR 12.00
			EUR 197.23	**EUR 118.73**

4.2 PCB Manufacturing

Manufacturer	Qty	3 PCB manufacture price	
PCB Pool	1	EUR 166.00	EUR 166.00

4.3 EMU Sensors

Element	Used Qty.	Unit Price (mean)	Price
ISSPRO Fuel sender	2	EUR 60.00	EUR 120.00

Hamlin Hall-Sensors	4	EUR 6.00	EUR 24.00
Westach EGT	2	EUR 38.00	EUR 76.00
VDO CHT	2	EUR 22.00	EUR 44.00
			EUR 264.00

4.4 Tools

Tool	Used Qty.	Unit Price (mean)	Price
MPLAB IDE	1	EUR 0.00	EUR 0.00
Explorer 16 DEV Board	1	EUR 60.00	EUR 60.00
MPLAB ICD2	1	EUR 120.00	EUR 120.00
MOLEX C-GRID connectors crimp tool	1	EUR 320.00	EUR 320.00
MOLEX MINI-FIT connectors crimp tool	1	EUR 280.00	EUR 280.00
			EUR 780.00

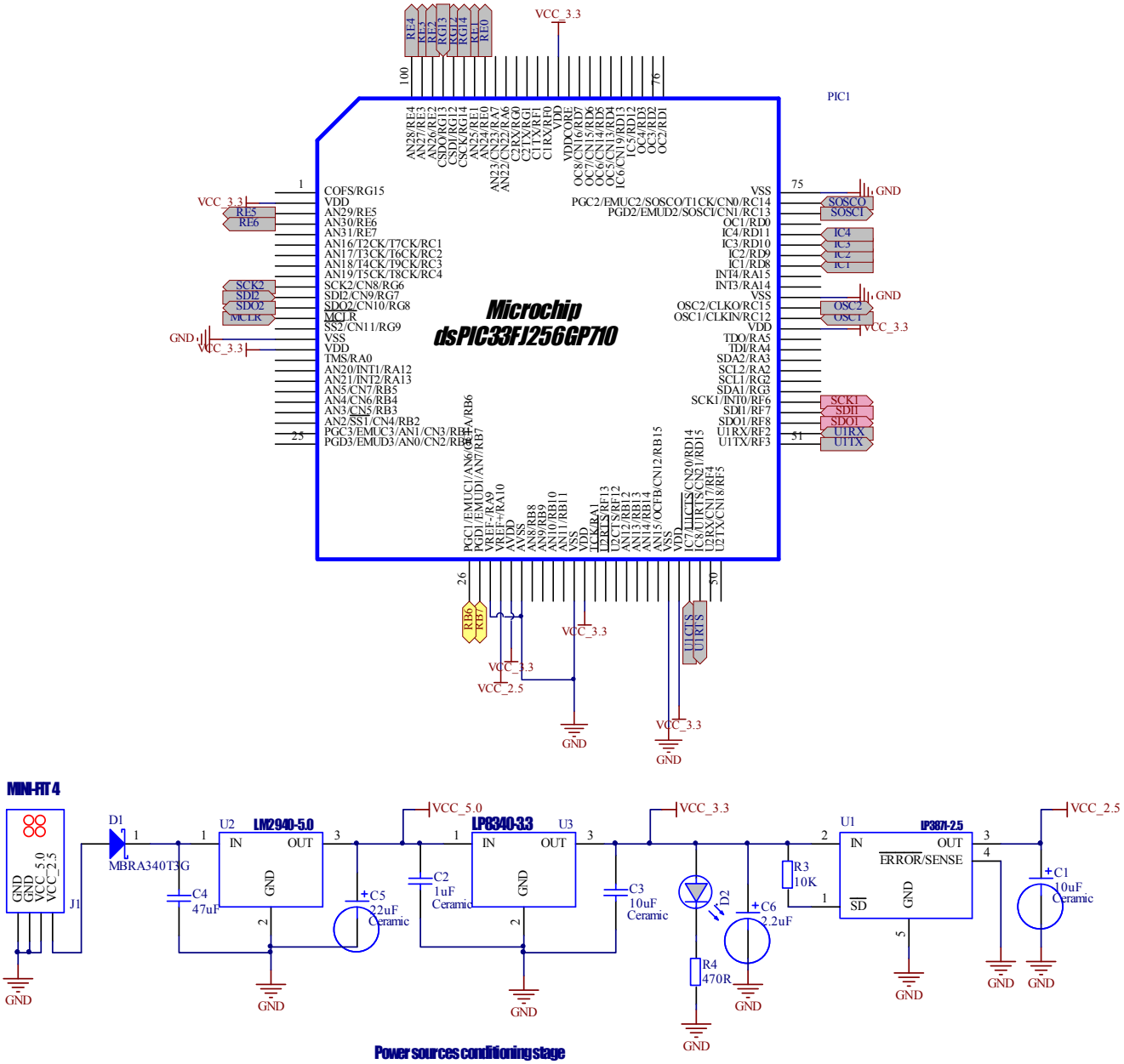
4.5 Conclusions

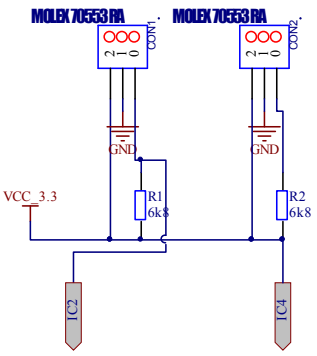
Total price for a single EMU printed circuit boards manufacturing is EUR 285.00

The sensors which will be used in EFMS implementation within the UAV Shadow have a price of EUR 264.00.

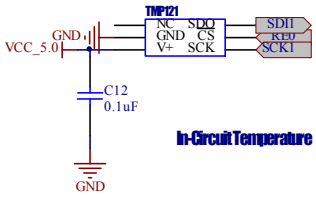
SECTION 5. COMPLETE ELECTRICAL SCHEMATICS

5.1 Module #1 Lower Board

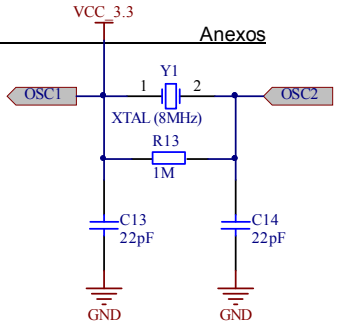




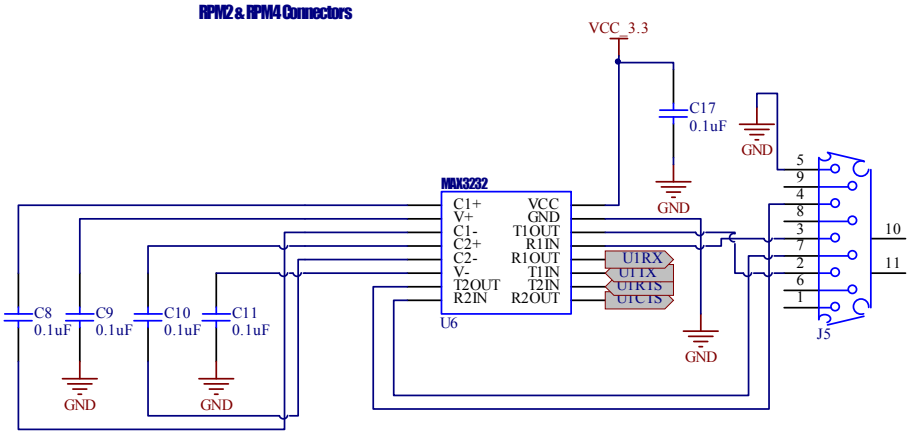
RPM2 & RPM4 Connectors



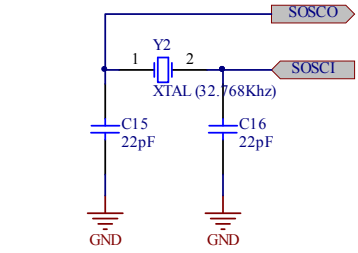
In-Circuit Temperature



Primary Clock Oscillator

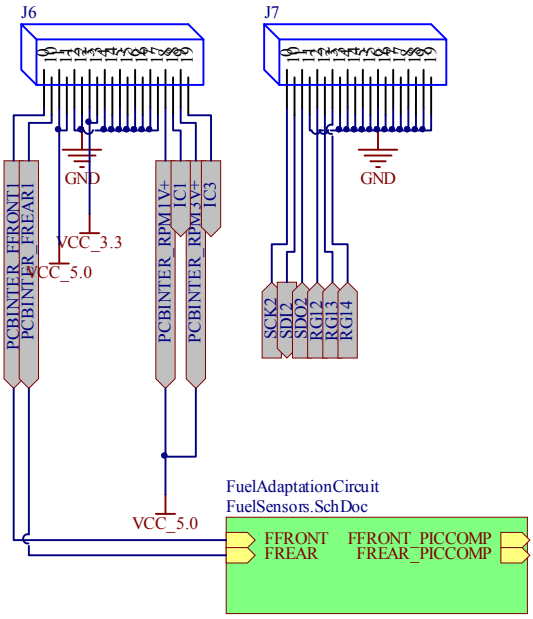


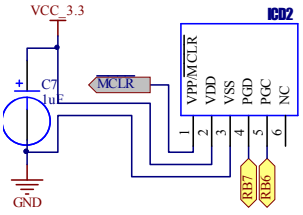
RS232 INTERFACE



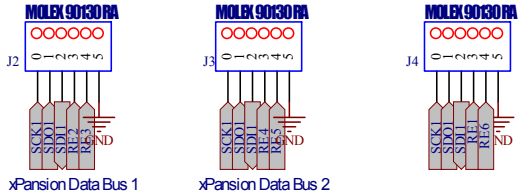
Secondary Clock Oscillator (Real Time CLK)

Connections to upper circuit



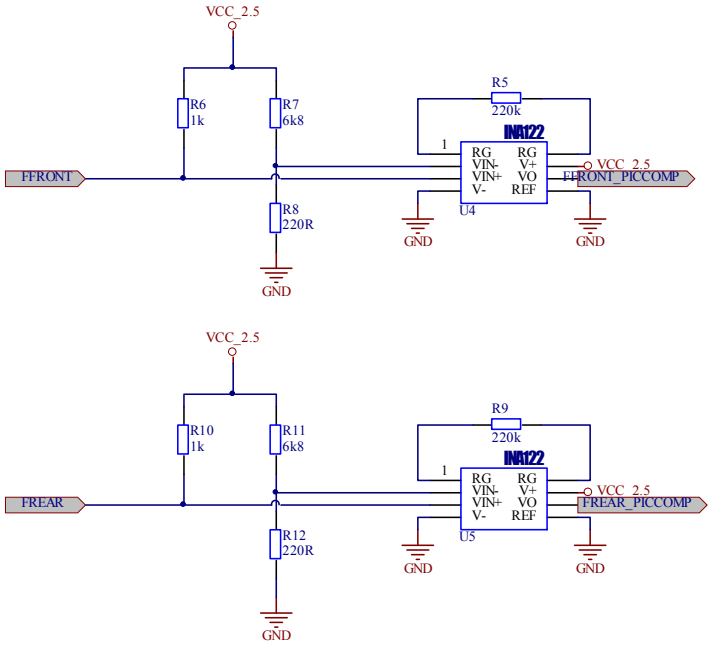


ICD2 Debugger/Programmer connector

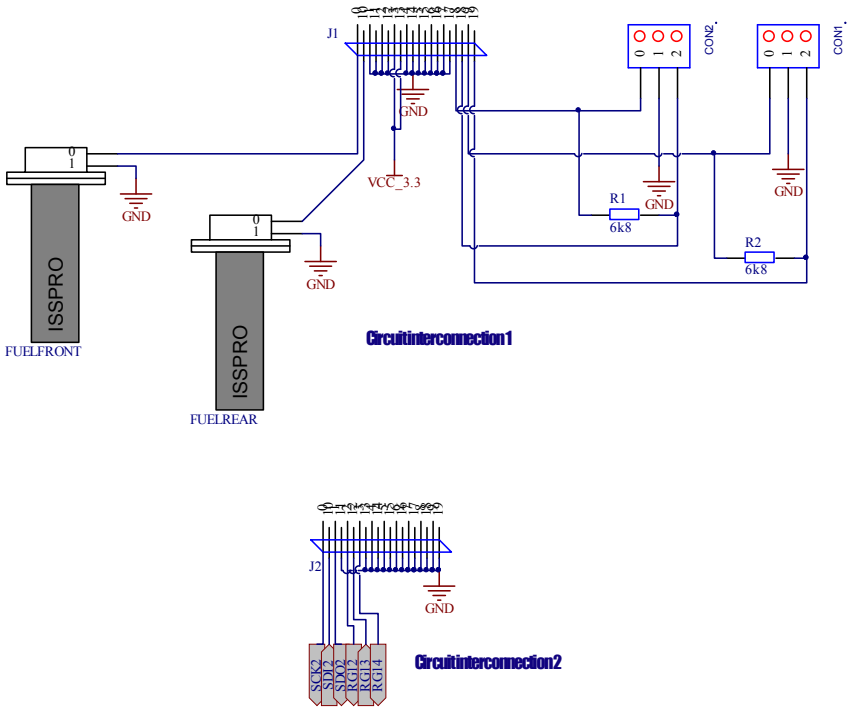


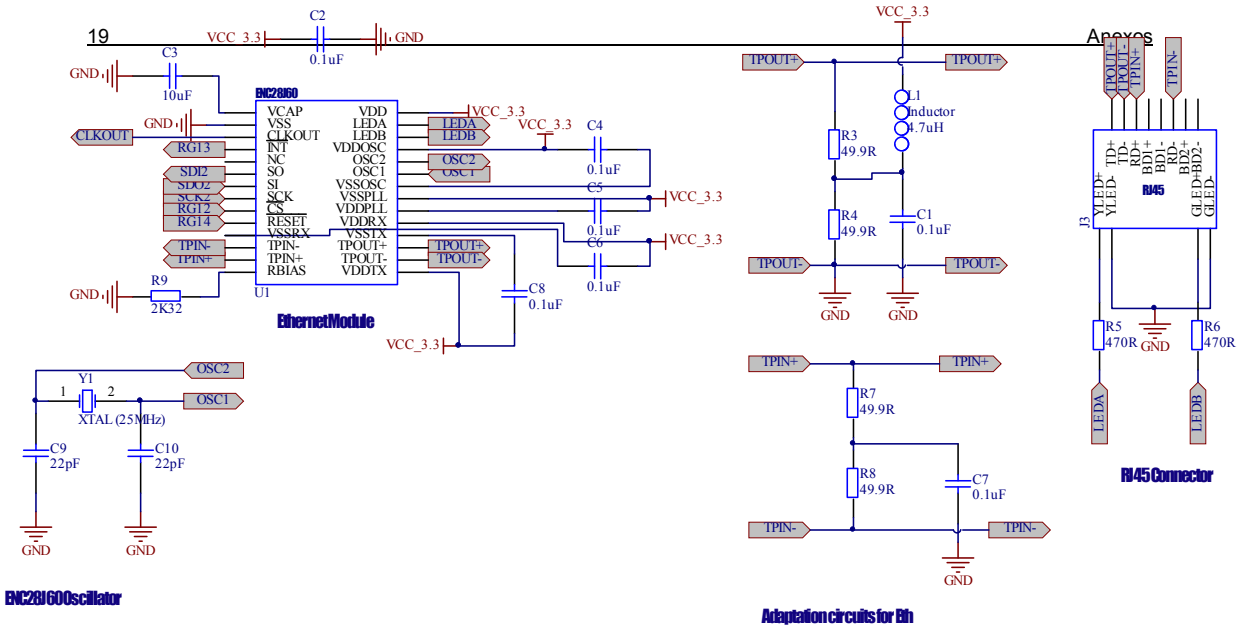
xPansion Data Bus 1 xPansion Data Bus 2

5.1.1 Fuel Acquisition Circuits (green box before)

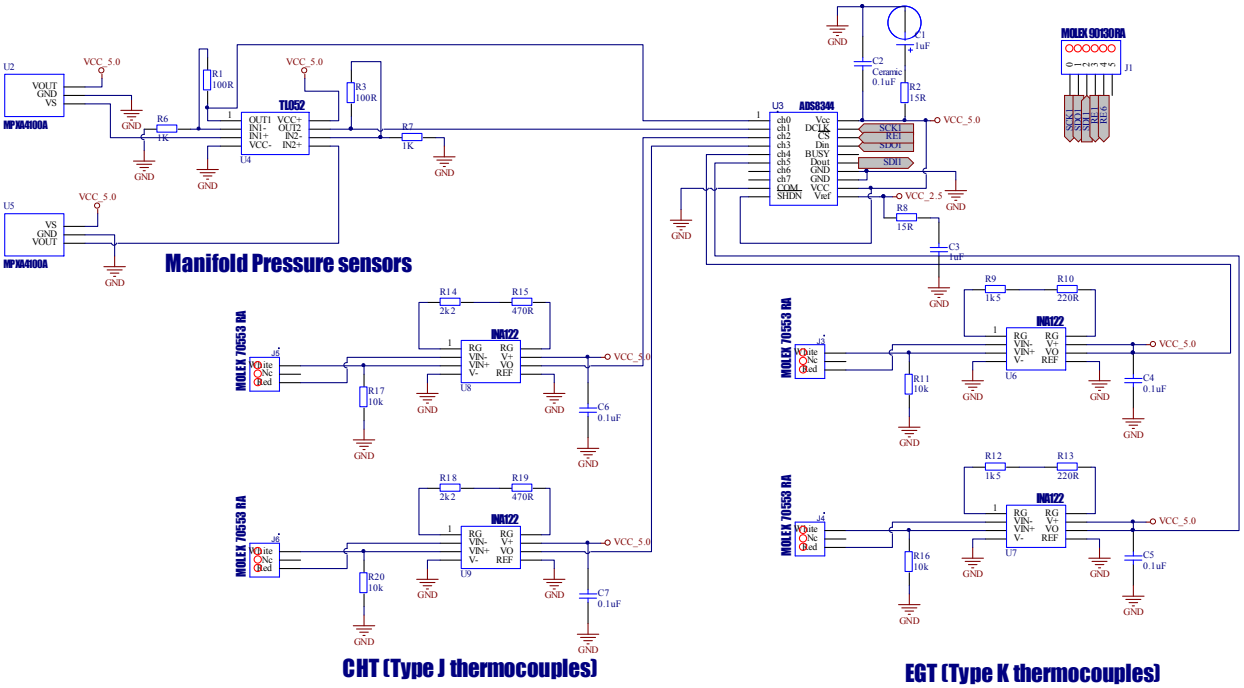


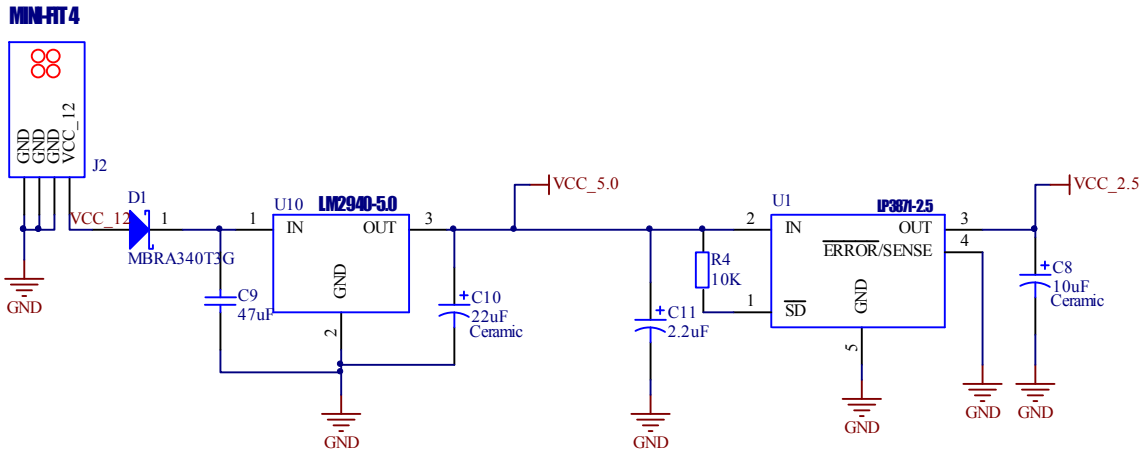
5.2 Module #1 Upper Board



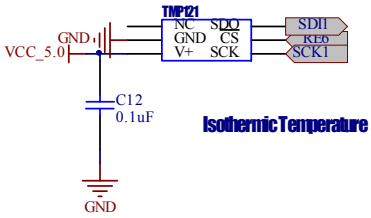

ENC28J60
EthernetModule
ENC28J60Oscillator
Adaptation circuits for Eth
RJ45 Connector
VCC_3.3
GND
TPOUT+
TPOUT-
TPIN+
TPIN-
LEDA
LEDB
OSC1
OSC2
XTAL (25MHz)
Inductor
4.7uH
RJ45

5.3 Module #2


Manifold Pressure sensors
CHT (Type J thermocouples)
EGT (Type K thermocouples)
MPX4100A
TL052
ADS8344
INA122
MOLEX 70553 RA
MOLEX 90130RA
VCC_5.0
VCC_2.5
GND



Power sources conditioning stage



Isothermic Temperature